TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

JOAKIM SUND

DETERMINING THE TURNING ANGLE AND SET SPEED OF STEER AND DRIVE UNITS IN VEHICLES FOR INDUSTRIAL USE

Master of Science Thesis

Examiner: prof. Jose L. Martinez Lastra
Examiner and topic approved by the Council meeting of the Faculty of Engineering Sciences on 31 May 2017

ABSTRACT

JOAKIM SUND: Determining the Turning Angle and Set Speed of Steer and Drive Units in Vehicles for Industrial Use
Tampere University of Technology
Master of Science Thesis, 79 pages, 10 Appendix pages
November 2017
Master's Degree Programme in Automation Engineering
Major: Factory Automation and Industrial Informatics
Examiner: Professor Jose L. Martinez Lastra

Keywords: Automated Guided Vehicles, Electric Vehicles, Kinematics, Programmable Logic Controllers, Self-Propelled Modular Transporters, Set Values, Wheeled Robots

The topic for this thesis was given by Ab Solving Oy, which is specialized in the design and manufacturing of complex, heavy load handling systems, ranging from single air bearing elements to complex Automated Guided Vehicle (AGV) systems. This thesis focuses on developing the software (SW) used in Solving wheeled vehicles. The current structure of the SW does not allow Solving to produce vehicle types that are asked for on the market.

Solving would like to base the vehicle control on Siemens Programmable Logic Controllers (PLCs), partly for economic reasons and partly for customer demand. The Siemens PLCs are currently used only in Solving air film movers, but should be expanded to be used in other vehicle types as well. The current SW supports the use of precisely two or four wheels, which must be positioned in a specific fixed relation to one another. The immediate challenge Solving is facing is that customers are asking for three-wheeled air film movers that are obviously not supported by the current solution. There are also inquiries for vehicles with more than four wheels, which Solving are also not able to fulfill today.

This thesis presents a new program structure that will enable Solving to produce vehicles with an unrestricted amount of wheels positioned anywhere within the construction of the vehicle frame. The outcome of this thesis consists of two types of Function Blocks (FBs). Software engineers at Solving can use the FBs to create any imaginable layout of the wheels on the vehicle. By using the outcome of this thesis, Solving will be able to meet the market demands that they are not able to meet today.

TIIVISTELMÄ

JOAKIM SUND: Pyörällisten Siirtolaitteiden Pyörien Kääntökulmien ja Asetusnopeuksien Määrittäminen ja Ohjaus
Tampereen teknillinen yliopisto
Diplomityö, 79 sivua, 10 liitesivua
Marraskuu 2017
Automaatiotekniikan diplomi-insinöörin tutkinto-ohjelma
Pääaine: Factory Automation and Industrial Informatics
Tarkastaja: professori Jose L. Martinez Lastra

Avainsanat: asetusarvo, asetusarvon laskenta, kinematiikka, ohjelmoitava logiikka, siirtolaite, sähköajoneuvo, vihivaunu

Tämän työn aiheen on antanut Ab Solving Oy, joka on erikoistunut räätälöityjen, raskaiden taakkojen siirtämiseen tarkoitettujen siirtolaitteiden suunnitteluun ja valmistukseen. Solvingin laaja tuotevalikoima ulottuu yksinkertaisista, yksittäisistä ilmatyynyelementeistä monimutkaisiin vihivaunujärjestelmiin. Tämän työn tärkein tavoite on kehittää pyörällisten siirtolaitteiden ohjelmistoa niin, että Solving pystyisi tuottamaan ajoneuvotyyppejä, joita he eivät pysty tuottamaan nykyisellä ohjelmistolla.

Solvingin tavoitteena, osittain taloudellisista syistä ja osittain asiakkaan vaatimuksiin perustuvista syistä on käyttää Siemensin ohjelmoitavia logiikoita kaikissa tuotteissaan. Solving käyttää tällä hetkellä Siemensin logiikoita ainoastaan ilmatyynyvaunuissa, mutta haluaisi laajentaa Siemensin käyttöä myös muihin ajoneuvotyyppeihin. Nykyinen Siemensiin perustuva ohjelmisto pakottaa Solvingin sijoittamaan siirtolaitteiden pyörät tarkasti määritettyyn asetelmaan ja pyöriä ei voi määrältään käyttää kuin kahta tai neljää. Monilla asiakkailla olisi tällä hetkellä tarvetta kolmipyöräisille ilmatyynyvaunuille, joita Solving selvästi ei voi tuottaa nykyisellä ohjelmistorakenteella. Lähitulevaisuudessa Solving haluaisi laajentaa tuotevalikoimaansa ja tarjota ajoneuvoja, joiden pyörien määrä on enemmän kuin neljä. Tähänkään tarkoitukseen nykyistä ohjelmistorakennetta ei voi käyttää.

Tämä työ esittää uuden ohjelmistorakenteen, millä Solving voi tuottaa ajoneuvoja, joiden pyörien määrä on rajoittamaton ja pyörien sijainti voidaan määrittää vapaasti mihin tahansa ajoneuvon fyysisen rakenteen sallimaan tilaan. Työn tuloksena on kaksi FB (Function Block) tyyppiä, joita Solvingin ohjelmistokehittäjät voivat käyttää tuottaakseen ajoneuvo-ohjelmistoja. Käyttämällä tämän työn ratkaisua Solving pystyy vastaamaan markkinalta tulevaan kysyntään, johon se ei ole kykyinen vastaamaan tänä päivänä.

PREFACE

The topic for this thesis was given to me during my first summer at Ab Solving Oy in July 2016. Because I at the time had a few courses left at Tampere University of Technology (TUT), the progress was quite slow in the beginning, during the academic year 2016-2017. Most of the work for this thesis was therefore done during the summer of 2017. In the beginning there was only a conceptual idea of what the exact outcome and goal of this thesis should be. It has been a long journey both time wise, knowledge wise and geographically getting from the conceptual idea to the implementation tested in real physical equipment. Research, mathematical analysis, design of the block structure and writing this document has been done mostly in buses, trains and airplanes during my commutes between Tampere, Pietarsaari, Nykarleby and Örebro, while the software and tests with the equipment were done at the Ab Solving Oy office in Pietarsaari. Working with this thesis has been very exciting, because testing with real products and seeing results in the physical form has always given me a special, rewarding feeling.

I want to thank everyone at Solving for the peaceful working environment and interesting discussions around the thesis. Thanks to Dick Edström for giving me the opportunity to be a part of the Solving team and special thanks to Torbjörn Södergård for valuable comments and supervision of this thesis. I also want to thank all fellow software engineers at Solving for great advice during this thesis and nice moments also outside the workplace.

I want to thank my teachers, friends and students at Tampere University of Technology (TUT) for encouraging me in my studies and interesting conversations related to different fields of technology. It has been five years of intense learning that I am sure will continue even after getting my degree. Special thanks to professor José L. Martinez Lastra for examining this thesis and valuable comments and suggestions not only in this thesis, but also in other courses at TUT.

I want to thank my friends and family back home in Nykarleby for supporting me and encouraging me in my studies and professional life. The final and greatest thanks belongs to my love Karoliina for motivation, genuine interest and continuous support.

Örebro, 03.11.2017

Joakim Sund

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AC	Alternating Current
AGV	Automated Guided Vehicle
B2B	Business-to-Business
CPS	Cyber-Physical-System
CVC600	Compact Vehicle Controller 600
DB	Data Block
DC	Direct Current
DO	Digital Output
E	East
EMT	Electric Modular Transporter
EPAS	Electric Power Assisted Steering
FB	Function Block
FBD	Function Block Diagram
HW	Hardware
ICR	Instantaneous Center of Rotation
km	kilometer
LOC	Lines of Code
LUT	Look-Up-Table
mm	millimeter
N	North
NE	Northeast
NW	Northwest
NaN	Not-a-Number
OB	Organization Block
PAS	Power Assisted Steering
PC	Personal Computer
PID Control	Proportional, Integral and Derivative Control
PLC	Programmable Logic Controller
PPU	Power Pack Unit
RC	Remote Controller
SCL	Structured Control Language
SPMT	Self-Propelled Modular Transporter
S	South
S/D	Steer/Drive
SE	Southeast
ST	Structured Text
SW	Software, Southwest
TIA Portal	Totally Integrated Automation Portal (Siemens programming tool)
VRML	Virtual Reality Modelling Language
W	West
WESN wheel	West-East-South-North wheel
4WS	Four-Wheel Steering

LIST OF SYMBOLS

Roman symbols:

d	distance	m
G	center of mass	-
max_{aj}	maximum analog joystick value	-
max_{vwh}	maximum allowed velocity	m/s
R	radius	m
R_{vc}	distance between the vehicle center and the ICR	m
R_{wh}	distance between a wheel and the ICR	m
v	velocity	m/s
v_{set}	set speed	m/s
v_{ref}	reference velocity	m/s
v_{vc}	velocity of the vehicle center	m/s
v_{wh}	wheel speed	m/s
x_{ICR}	x-coordinate of the ICR	m
x_{LJ}	the horizontal analog value of the left joystick	m
x_{LM}	x-coordinate of the leftmost wheel	m
x_{RJ}	the horizontal analog value of the right joystick	m
x_{RM}	x-coordinate of the rightmost wheel	m
x_{VF}	x-coordinate of the vehicle front	m
x_{VR}	x-coordinate of the vehicle rear	m
x_{wh}	x-coordinate of a wheel	m
y_{ICR}	y-coordinate of the ICR	m
y_{LJ}	the vertical analog value of the left joystick	m
y_{RJ}	the vertical analog value of the right joystick	m
y_{VF}	the y-coordinate of the vehicle front	m
y_{VR}	the y-coordinate of the vehicle rear	m

Greek symbols:

α	desired heading	deg
θ	turning angle	deg
ω_{ref}	angular velocity	rad/s

1. INTRODUCTION

Vehicle types that are familiar to most people include cars, buses, trucks and heavy machinery such as tractors and excavators. Industrial vehicles on the other hand are vehicles that are most likely not familiar to the majority of people. These are vehicles that are used in environments such as factories, docks, offshore applications and in non-recurring tricky situations. An industrial vehicle can be equipped with for instance 16 wheels that must all be able to turn at the same time, while a common, ordinary car is equipped with just four wheels, where only the two front wheels turn. The commissioner of this thesis, Ab Solving Oy produces industrial vehicles ranging from simple, single air bearing elements to complex Automated Guided Vehicle systems (AGV systems). The main point of focus for this thesis are the wheeled industrial vehicles made by Solving.

1.1 Motivation and Justification

Solving is under a lot of pressure to continuously develop their products and increase overall efficiency to keep up with the competition. Handling systems that should be able to handle heavier, larger and more complex loads than before are asked for on the market every day. The products and Solving itself must therefore keep evolving to be able to provide customers with wanted products. In the scope of this thesis Solving needs to develop specifically the vehicle software, in order to be able to sell and manufacture products they are not able to do today. The vehicle software needs to be developed also such that Solving will be able to make products that will be interesting in the near future. The particular problem this thesis addresses is currently not allowing Solving to fulfill all inquiries coming from customers.

1.2 Problem Statement

The vehicle software Solving is using in the Siemens Programmable Logic Controller (PLC) based vehicles today allow for precisely two or four wheels, positioned in a specific fixed relation to one another. Normally one vehicle contains two wheels and there is an option to connect two vehicles together either side-by-side or end-to-end. Two or four wheels positioned in this particular layout is obviously not enough for providing the required support for the heavy loads the vehicles must be able to handle. The size and weight of the load, can easily require 16 wheels and sometimes even more.

The immediate challenge Solving is facing is that customers are asking for air-film movers with three wheels, which are clearly not supported by the current vehicle software.

Just as for the two-wheeled air-film movers there should be an option to connect two three-wheeled vehicles together, which means that the software should be able to handle not only three wheels, but also six wheels. The vehicle software in the air-film movers should also be implementable in other vehicle types. It is for instance predicted that there will be a demand for multi-wheeled vehicles, not using air bearings in the near future. The amount and position of wheels in a multi-wheeled vehicle is not specified in any way. Multi-wheeled vehicles with more than four wheels are clearly also not supported by the current vehicle software.

1.3 Objectives

The work of this thesis is part of a larger development project at Solving. The goal of the project is to create better consistency in the production and allow for use of cheaper components in the products. One of the main goals for achieving better consistency is to base the choice of vehicle controller on only one controller brand, namely Siemens PLCs. Solving is today using two vehicle controllers, the Kollmorgen CVC600 and Siemens ET200SP. The support for Siemens components is better and the Siemens components are cheaper in comparison to the Kollmorgen components.

The main objective of this thesis is to develop the vehicle software used in Siemens controlled vehicles, so that Solving will be able to handle more customer inquiries with the Siemens based control. The vehicle software should after this thesis no longer be an obstacle for creating the aforementioned vehicle types that Solving is not able to produce today.

1.4 Outline

The purpose of this document is to present research, a proposal solution, practical work and outcome of the thesis. The practical work and this document follows the very common top-down design principle, meaning that the abstraction level is quite high in the beginning of the document and very low at the end of the document. The reader of this thesis will form an understanding of industrial vehicles and the system used in this thesis, before moving on to a more detailed level of the outcome of this thesis. Different types of vehicles, Solving's products and requirements on the proposed solution will be presented in chapter 2. The system in question as well as mathematical analysis and derivation of algorithms and equations needed for the proposed solution are presented in chapter 3. A block structure that will be used for the implementation of the proposed solution will be presented in chapter 4 after which test results are presented in chapter 5. Analysis of the results, outcome, future work and conclusions are presented in the final chapter.

2. LITERATURE AND INDUSTRIAL PRACTICES REVIEW

The vehicles concerning this thesis operate in a different environment and behave differently compared to common vehicles that can be seen in traffic and at construction sites. They operate in industrial environments and must be able to move in very challenging places. These vehicles are known as industrial vehicles and include vehicle types such as Automated Guided Vehicles (AGVs) and Self-Propelled Modular Transporters (SPMTs). They are powered with a motor or several motors like combustion engines, hydraulic motors, pneumatic motors, electric motors and combinations of these. There is usually one or several powerful drive motors that cause the vehicle to move, while some less powerful motors are responsible of steering the vehicle. In this chapter the operating principle of different types of vehicles and the behavior of the wheels or chains on the vehicle is presented. The idea of the structure of this chapter is to start from vehicle types that should be somewhat familiar to most people and step by step zoom in on a more detailed level of the complex vehicle types. The outline of this chapter is presented in figure 1.

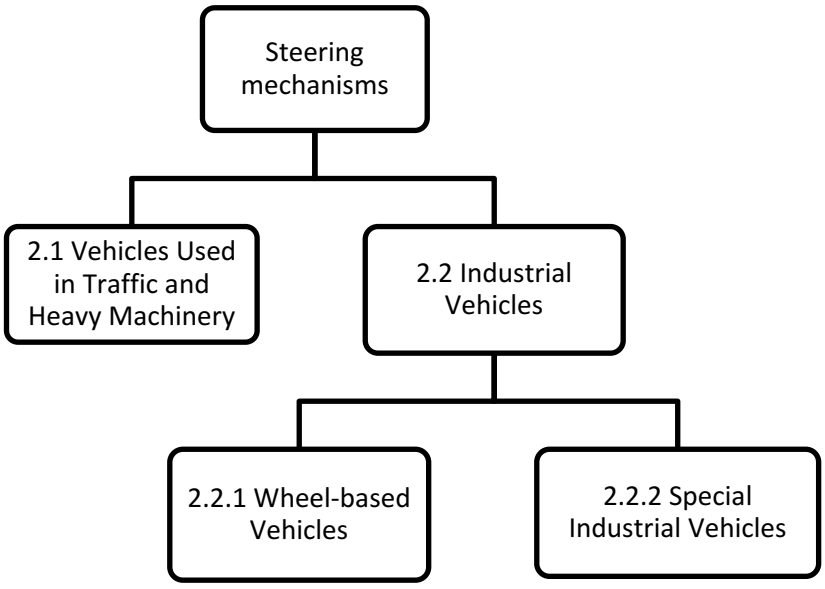

Figure 1. *Outline of chapter 2.*

The outline presented in figure 1 is followed by section 2.3, where the vehicle types produced by Solving and the affect this thesis will have on them, are presented. At the end of the chapter it should be clear how each steering mechanism differ from one another and what requirements the vehicle types made by Solving set on the work of this thesis.

2.1 Steering Mechanisms in Vehicles Used in Traffic and Heavy Machinery

The structure of the steering mechanism in bicycles and motorcycles is really simple. There is a handlebar that is mounted on the front wheel creating a physical connection between the front wheel and the handlebar. Because the handlebar and the front wheel are physically linked, moving the handlebar will result in an equal movement of the front wheel. The rear wheel on the other hand is fixed in the vertical direction of the frame, but rotates by the power of a human or an engine. The front wheel steers the bicycle or motorcycle, i.e., determines the direction of the vehicle, while the rear wheel causes the vehicle to move.

The so called kinematic bicycle model presented in figure 2 is commonly used as a starting point for kinematic calculations (Kong *et al.* 2015; Snider 2009; Giancarlo *et al.* 2009; Jazar 2008; Lakkad 2004). When a bicycle is turning, the front wheel, center of mass (G) and the rear wheel will all respectively create circles with different radii around the Instantaneous Center of Rotation (ICR). When the normal lines to all the wheel planes intersect at the ICR, the vehicle is moving in a slippage free manner. This condition will be handled for different types of vehicles several times in this thesis. Although the vehicle types will be more complex, the geometrical principle is the same for the more complex vehicle types as for the kinematic bicycle model.

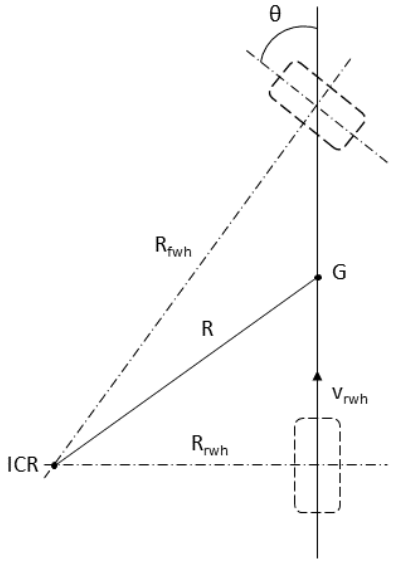

Figure 2. *The kinematic bicycle model (modified from Giancarlo et al. 2009).*

Bicycles and motorcycles require the driver to have a sense of balance in order to keep the vehicle upright. If the driver leans too far to one side, the vehicle will tip over. A vehicle with four wheels instead of two is physically stable and will stay upright even when it is not moving. The least complex stable vehicle from a theoretical point of view, is the four-wheel parallel steered vehicle, which in practice could be achieved by welding

two bicycles or motorcycles and their handlebars together side-by-side such that using one of the handlebars would cause both the front wheels to turn the same amount. In other words such a four-wheel type vehicle could be modelled by adding another bicycle next to the existing bicycle in figure 2. Both front wheels would then always have the same turning angle, i.e., the angle θ in figure 2. While such a setup is theoretically very simple it is in direct conflict with the slippage free driving condition. If the turning angle of both front wheels is the same, all the normal lines to the wheel planes, i.e., wheel axes no more intersect at the same point. In practice it means that the parallel steering will cause a scenario, where the wheels will try to outrun each other. Because the wheels are attached to the vehicle body, they will however not be able to outrun each other and instead they will slip or jump, which will cause damage to either the wheels or the surface the vehicle is moving on.

In some applications damaging the wheels or the surface the vehicle is moving on are factors that does not need to be accounted for. Heavy equipment such as excavators, skid-steered loaders, forest machines etc. operate in conditions, where operational reliability is important and the surface they are moving on is usually not damageable. Soft soil for instance, require large wheels or chains in order to create a large contact area such that the vehicle will not get stuck. Usually there is one chain or a set of wheels on the left side and another chain or set of wheels on the right side of the vehicle. One motor or a set of motors are connected to the chain on the left side and another set of motors are connected to the chain on the right side. The chains and thus the vehicle is operated with two joysticks. One joystick controls the speed of the chain on the left side and the other joystick controls the speed of the chain on the right side. If both joysticks are given the same amount of throttle, the vehicle will move straight. If on the other hand, the joysticks are given different amounts of throttle the vehicle will turn to the side that is given less throttle. This type of steering is called skid-steering and can for instance be found in skid-steered loaders made by Caterpillar[1]. As the name implies the slippage, also known as skidding, is a part of the steering concept. Although slippage is occurring, the strong chains on the vehicle are not significantly damaged by the soft soil they are usually moving on.

On public roads and traffic in general, damage on the roads and wheels should be avoided. This makes the parallel steering and skid-steering concepts impractical. George Langensperger introduced the geometric condition for slippage free driving for four-wheeled vehicles in 1816. This condition has later been named after his patent agent Rudolf Ackerman (Jazar 2008). The Ackerman condition supports the earlier stated fact that slippage free driving is accomplished when the normal lines to all wheel planes intersect at a common point. The most common steering mechanism following the Ackerman condition is the trapezoidal steering mechanism. For the trapezoidal steering mechanism to work, the arms A and B need to point towards point C as shown in figure 3. Only then the axes of

[1] http://www.cat.com [Accessed: 26.09.2017]

the front wheels will intersect at a common point. A detailed analysis of the trapezoidal steering mechanism is presented in (Jazar 2008).

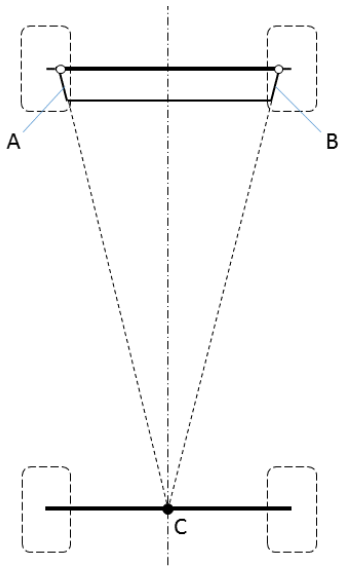

Figure 3. *The trapezoidal steering mechanism following the Ackerman steering condition for slippage free driving (Jazar 2008).*

The Ackerman steering geometry and particularly the trapezoidal steering mechanism is the steering mechanism that is used in cars. Although the response of the steering wheel can feel different in old cars compared to modern cars, they all follow the same geometrical principle. In old cars, turning the steering wheel requires a lot of effort, because it is physically connected to the displaceable link between arms A and B. In some old cars the steering is assisted with a hydraulic cylinder, which makes it easier to turn the steering wheel. The hydraulic cylinder was however seen inefficient, due to drag and parasitic power losses caused by the hydraulic pump. Electric motors are therefore nowadays more commonly used in the Power Assisted Steering (PAS) systems. There are several benefits of using an Electric Power Assisted Steering (EPAS) system compared to a hydraulic one. The electric system is for instance more energy efficient, cleaner, more accurate and is more addressable for programming. The PAS is an important part of the behavior of the steering, although it does not affect the geometrical relationships of the wheels in a vehicle. (Knowles 2011)

In addition to the presented steering mechanisms so far there are two more that affect the orientation of only two wheels, i.e., the articulated axle and the articulated frame steering. The axle articulated steering concept is a simple steering concept, where the axle connecting the front wheels can be rotated while the wheels themselves remain straight (Lakkad 2004). The axle articulated steering concept is commonly used in for instance soapbox cars (gravity racers), because of the simple construction. All that is needed is a joint in the axle between the two front wheels. Please note that axle articulated steering does not interfere with the slippage free driving condition as the normal line to both the front wheels are aligned and always have a common intersection point with the normal line to

the rear wheels. Although the articulated axle steering can result in very tight turns, it is not really applicable in vehicles such as cars for instance, as a rotating axle would require a lot of space and the construction of car chassis would need to be entirely redesigned.

In articulated frame steering, the part of the vehicle that is physically articulated is the vehicle frame. By turning both the wheels and the frame, the vehicle can undertake extremely steep curves, which make the articulated frame steering suitable for applications where the vehicle is required to make tight turns. Valtra is a Finnish company that produces frame articulated tractors[2]. A frame articulated Valtra tractor has five different steering modes. Mode A follows the traditional Ackerman condition with the chassis locked in idle position and only the front wheels cause the turning. In mode B both Ackerman type steering and the articulated frame is used. The wheels and the chassis turn simultaneously so that the condition for slippage free driving is maintained. Mode C is called parallel track, intended especially for agricultural applications and is excellent for reducing soil compaction. In mode D, slippage free steering is accomplished using only the articulated frame. In mode E the turning angle of the front wheels and the angle of the frame can be controlled separately, unlike mode B, where the frame and wheels are turned simultaneously. The steering modes A-E are presented in figure 4.

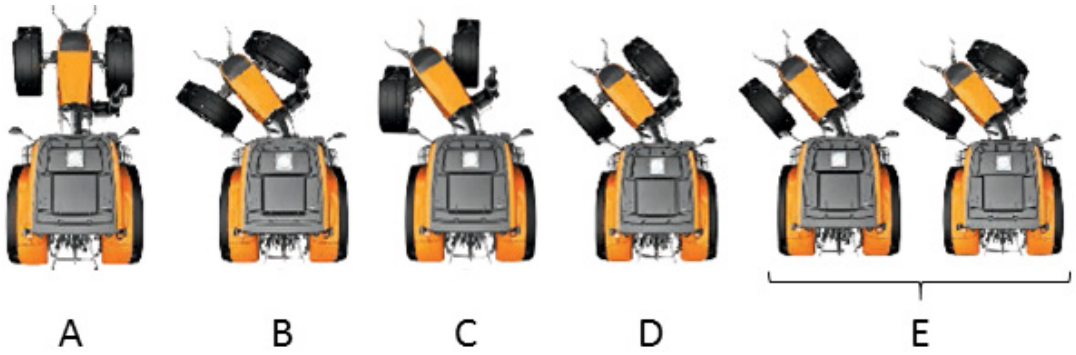

Figure 4. *A: Front wheel steering only, B: Combined front wheel steering and frame articulation, C: Parallel track, D: Frame articulation only, E: Separate front wheel steering and frame articulation.*

While the articulated axle or frame steering would clearly reduce the turn radius and improve the maneuverability of other vehicle types than tractors, the articulation require a certain amount of space, which would mean that other features such as the comfort level of the vehicle or the size of the vehicle would suffer. A better way to achieve a smaller turning radius is to turn both the front wheels and rear wheels at the same time instead of physically articulating a part of the vehicle body. On a four-wheel vehicle such as a car, this type of system is called a Four-Wheel Steering (4WS) system. Audi (Audi 2017), Nissan (Nissan 2016) and BMW (BMW 2016) among others all have their own version of the 4WS system. Although 4WS differs in some ways depending on the manufacturer,

[2] http://www.valtra.com/articulated-tractors.aspx [Accessed: 30.11.2016]

the main principle is the same. 4WS is achieved by installing a trapezoidal steering mechanism on the front axle as well as on the rear axle. Typically, point C is not located in the middle of the vehicle, but a bit above or under the rear axis as illustrated in figure 5, which means that during a turn, the front wheels will turn more than the rear wheels.

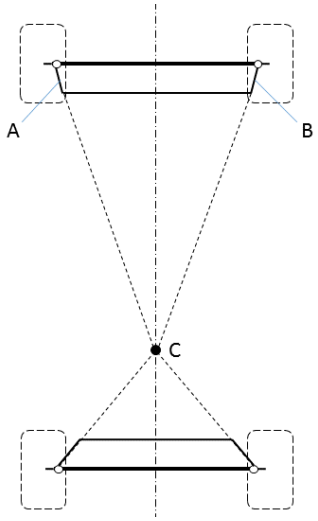

Figure 5. *Four-Wheel Steering (4WS) achieved by trapezoidal steering mechanisms on the front and rear axles (modified from Jazar 2008).*

The amount of wheels and axles on a bus or truck vary significantly depending on the model and application. The vehicle can be built just as a car, i.e., four wheels and two axles, where only the front wheels turn or all wheels turn as in the 4WS systems. There can also be additional axles intended for support of heavy loads. The wheels on the extra axles are sometimes fixed such that they cannot be turned and sometimes the axle is equipped with a trapezoidal steering mechanism similarly as in 4WS systems, maintaining the Ackerman condition for slippage free driving. Adding a fixed third axle to the 4WS system is no problem as long as the axle is located exactly at the intersection point C, as for instance in some Mercedes[3] trucks. In vehicles where there is at least two fixed axles on the other hand, the tires on at least one of the axles will be subject to an enormous amount of wear, due to a direct conflict with the slippage free driving condition. One way to reduce wear of the tires on a bus or a truck like this is to lift the extra axles, whenever possible. This technology is used among others by Volvo[4]. The size of buses and trucks as well as the load they are handling may in addition to the extra axles, require an extra powerful PAS, consisting of both a hydraulic and an electric motor. Such PAS systems are also used on Volvo trucks.

One of the first things mentioned in this section is that the wheels on a vehicle will all create circles around a common point called the ICR and that these circles will all have different radii. While it is very essential that the normal lines to all wheel planes intersect

[3] https://www.mercedes-benz.co.id/content/indonesia/mpc/mpc_indonesia_website/enng/home_mpc/truck_home.html [Accessed: 27.09.2017]
[4] http://www.volvotrucks.com/en-iq/home.html [Accessed 27.09.2017]

at the ICR, it makes no difference if the relation between the speeds of the wheels is incorrect. If the direction of the wheels satisfy the Ackerman condition of slippage free driving, but the speeds don't add up, the wheels will still slip or jump. Consider for instance a general case of a four-wheel vehicle driving according to the Ackerman condition. If all the wheels were to have the same speed some of them would try to outrun some of the other wheels. The differential is a mechanical component that allow the driving wheels to run at different speeds, what for slippage can be avoided. Just to be clear, the differential does not apply more power to one wheel and less to another, it only divides the power to the wheels such that it allows the wheels to run at different speeds.

The main components of the differential are the transmission shaft, driving pinion, crown wheel, sun gears, the planet pinion and the half-shafts going to the left and right wheel. The power of the engine is transferred through the transmission shaft to the crown wheel via the driving pinion. The crown wheel that is not attached to the half-shaft going to the wheels and is able to rotate without being affected of the half-shafts, rotates the planet pinion. When the vehicle is moving straight, the rotational movement of the crown wheel and the planet pinion, moves the sun gears of both wheels at the same speed as the crown wheel. When the vehicle is turning, the planet pinion allows the sun gears to rotate at different speeds. This way the wheels are allowed to rotate at different speeds although the wheels have the same source of power. The differential and all the mentioned parts are presented in figure 6.

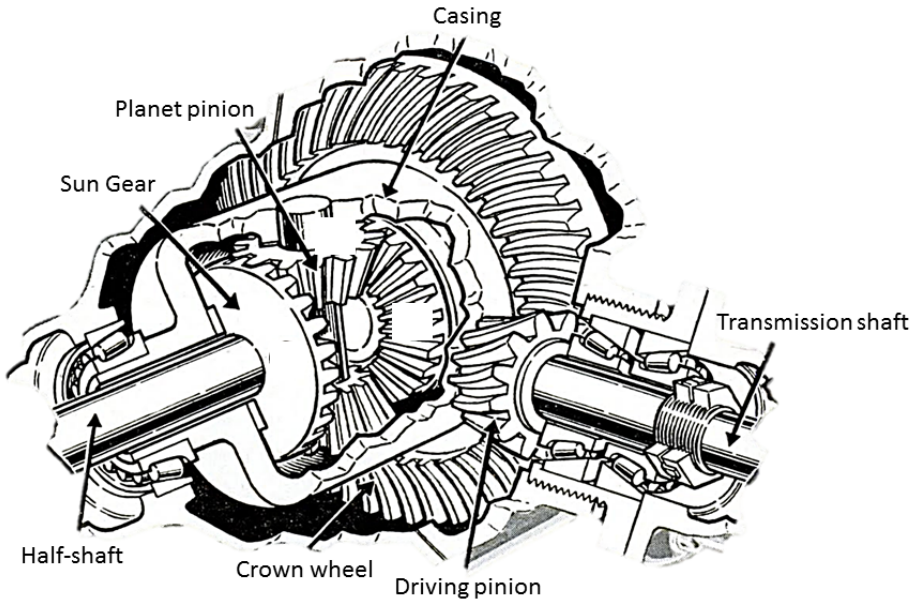

Figure 6. *The internal construction of the differential (modified from Gitchens 1946).*

The construction and operating principle of the differential is simple, but might be a bit difficult to understand. The point here is not to explain all the details of the differential, but to show that in difference to the mechanisms that will be presented in the following sections it is fully based on mechanical components only.

2.2 Steering Mechanisms in Industrial Vehicles

The vehicles presented in the previous section should have been somewhat familiar from other context to the majority of people. The vehicles can for instance be seen on the public roads or at construction sites. All types of vehicles and every detail of them can obviously not be handled in this thesis, due to the large amount of different vehicle types and details regarding the vehicles. Rear wheel steered fork lifts, where for instance not and will not be brought up as the steering concept in those follow the same principles as the steering concept in front-wheel steered vehicles.

The requirements on a vehicle operating in industrial environments are completely different compared to requirements on more traditional vehicles. The velocity of the industrial vehicles are normally much smaller and the behavior of the movements of the industrial vehicles is much more flexible than vehicles used in normal traffic. The industrial vehicles often require some amount of programming in difference to non-industrial vehicles. Industrial vehicles are usually not sold for private use, they are sold Business-to-Business (B2B) or rented for non-recurring tricky operations. The details of the steering mechanisms are therefore for the most part business secrets, which is why the information for this section had to be combined from several different sources.

2.2.1 Steering Mechanisms in Wheel-based Industrial Vehicles

When large, heavy or otherwise difficult loads need to be moved, Self-propelled trailers or Self-Propelled Modular Transporters (SPMTs) are preferably used. These types of transporters can be connected to a truck or used just as is. A modular transporter can not only be used in single mode, individually, but can also be operated in tandem, i.e., several transporters can be connected together to move in unison. In this section information about self-propelled vehicles and Automated Guided Vehicles (AGVs) has been gathered from the field's leading manufacturers such as Scheuerle[5], Goldhofer[6], Cometto[7], Kollmorgen[8], Rocla[9] and (Solving 2017). Further referencing of these companies is done only by mention. An example of several interconnected SPMTs made by Scheuerle is presented in figure 7. It can be difficult to see from the figure, but self-propelled trailers and SPMTs move in the same slippage free manner as presented earlier, which means, as also stated earlier that the normal lines to all the wheel planes intersect at a common point, the ICR.

[5] https://www.scheuerle.com/ [Accessed 18.12.2016]
[6] http://www.goldhofer.de/ [Accessed 18.12.2016]
[7] http://www.cometto.com/en/ [Accessed 18.12.2016]
[8] http://www.kollmorgen.com/en-us/home/ [Accessed 18.12.2016]
[9] http://www.rocla-agv.com/ [Accessed 18.12.2016]

Figure 7. *Several Self-Propelled Modular Transporters (SPMTs) linked together and working in unison to move a large crane.*

Each wheel in an SPMT is equipped with a drive motor, hence the name “self-propelled” and usually also a steer motor. The drive and steer motors are powered by one or several Power Pack Units (PPUs), which can be found at one end of the SPMT, at the left side of figure 7 for instance. The individually controlled drive motors on each wheel make the SPMTs very flexible. One way to ensure the correct turning angles and speeds of each wheel is to control the speed of each drive motor programmatically, for controlling the wheel speeds, and use a mechanical linkage similar to the trapezoidal steering mechanism presented earlier, for controlling the turning angles of the wheels. The mechanical linkage can be configured such that it turns all wheels or just the wheels in front of the rear axle or just the wheels under the front axle. Nevertheless so that once the mechanical linkages has been installed, the configuration cannot be changed. Such solutions that are based only on mechanical parts seem to be quite uncommon nowadays, because the mechanical construction cannot be changed and limits the movements of the vehicle. Controlling the speed and turning angle of each wheel separately is a much more common and flexible solution.

Controlling the speed and turning angle of a wheel require wheels that are equipped with drive and steer motors. The entirety consisting of a wheel and a drive motor is called a drive unit, while the entirety consisting of a wheel and a steer motor is called a steer unit. If there is both a steer and drive motor connected to the wheel, the unit is called a steer/drive unit. In practice, it is assumed that a wheel is equipped with both a steer and drive motor and the unit is usually called for simplicity just drive unit, although the unit is clearly a steer/drive unit. The terminology might be a bit confusing, but from this point forward a drive unit is assumed to consist of a wheel, a steer motor and a drive motor, unless stated otherwise.

Because drive units are able to both drive and steer, there is no need to have physical connections between the wheels. The control of the speed of the steer motor and drive motor must on the other hand be carefully programmed in order to continuously maintain the Ackerman condition of slippage free driving. Because there is no mechanical lock on the wheels, it is possible to change even the way of steering the vehicle. The vehicle can be driving in a configuration where all wheel axes intersect at a point in-line to the vehicle center and suddenly the steering can be changed such that the wheel axes are parallel instead of intersecting at a common point and so on. These different ways of steering an SPMT are called steering programs or sometimes drive modes, of which the steering program, where all normal lines to the wheel planes intersect at a point in-line with the vehicle center is called the “counter-phase” steering program and turning the wheels such that the normal lines to the wheel planes remain parallel is called the “crabwise” steering program.

While using a drive motor and a steer motor for each wheel is clearly the best solution for achieving a vehicle with several different steering programs, an attempt of using only mechanical parts has been done in (Bhishikar *et al.* 2014). A mechanical construction like that one is clumsy to use and unnecessarily complex, which is probably why such constructions cannot be found in any real application and why it will not be pursued here. The mechanical design of the vehicle is although complex, quite interesting and in the scope of this thesis worth mentioning, because readers of this thesis might be wondering, whether such vehicles exist or not.

It should be quite clear why one would like to be able to use different steering programs in order to change the behavior of the vehicle movements. It is for instance not possible to drive straight in the direction of 90° using any of the traditional front wheel and rear wheel steering programs, but fully possible using the “crabwise” steering program. The use for different steering programs should be intuitive, but if necessary the interested reader can see (Singh *et al.* 2014) for a more detailed analysis of the benefits of using several steering programs.

The most basic steering programs that resemble the steering concept in front wheel steered and 4WS cars are the “front wheel steering”, “rear wheel steering” and “counter-

phase steering" programs. In difference to a car, the amount of wheels on an industrial vehicle is not limited in any way. The vehicle in figure 8 has for instance eight wheels, but any other amount of wheels is also possible for SPMTs.

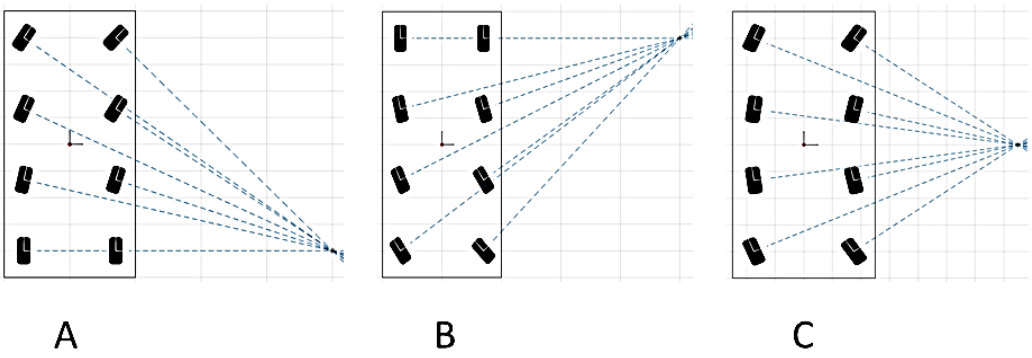

Figure 8. *A: Front wheel steering, B: Rear wheel steering, C: Counter-phase steering.*

Another steering program that was in some extent presented earlier is the "differential drive mode". The differential drive mode is quite similar to the steering mechanism in skid-steered loaders presented earlier. Usually one joystick controls the speed of the right side of the vehicle and another controls the speed of the left. Although the vehicle in figure 9 only has two wheels to better show for the connection between the skid-steering and differential drive mode, there is no restriction to the amount of wheels as long as they are all aligned with the vehicle center. If a wheel would be added to the vehicle center of the vehicle in figure 9 and only the right side would be given a velocity for instance, the wheel on the right side would have the largest velocity, the wheel at the vehicle center would have a bit smaller velocity and the velocity of the wheel to the left would be zero.

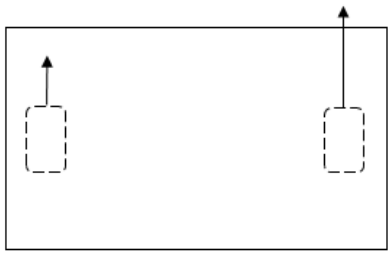

Figure 9. *Differential drive mode steering program.*

The steering programs in figure 8 and 9 should in some extent feel familiar, because they are quite similar to the steering concepts of non-industrial vehicles. In addition to these steering programs there are two steering programs that are not usually seen on other vehicles than industrial vehicles. These steering programs are the "crabwise" and the "rotation mode" steering programs. The crabwise steering program and the rotation mode are very useful in industrial applications, where the space is many times quite small and the movements of the vehicle need to be as flexible as possible. The crabwise steering program is for instance used in situations, where the position of the vehicle is wanted to be

changed, but the orientation is liked to be kept the same. When the position is on the other hand liked to be kept the same and only the orientation of the vehicle is wanted to be altered, the rotation mode is used. The crabwise and rotation mode steering programs are presented in figure 10, where it should also be quite clear what the difference between the two steering programs is.

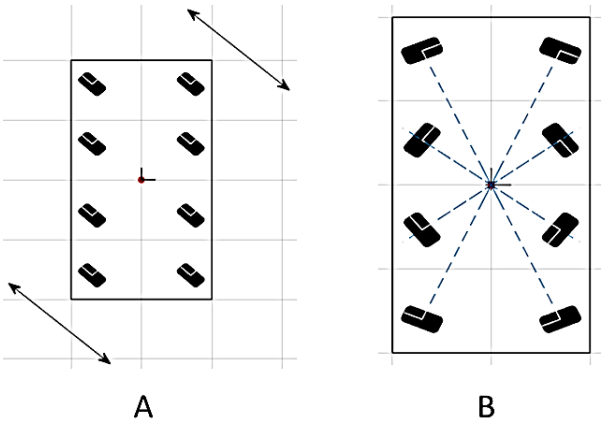

Figure 10. *A: Crabwise steering, B: Rotation mode.*

Some SPMT types also has an option to change the orientation of the vehicle from the lengthwise direction to the crosswise direction. The most basic steering programs would then change and look like presented in figure 11. The range of the turning angle of each drive unit must of course be large enough to be able to change the orientation of the vehicle from lengthwise direction to the crosswise direction.

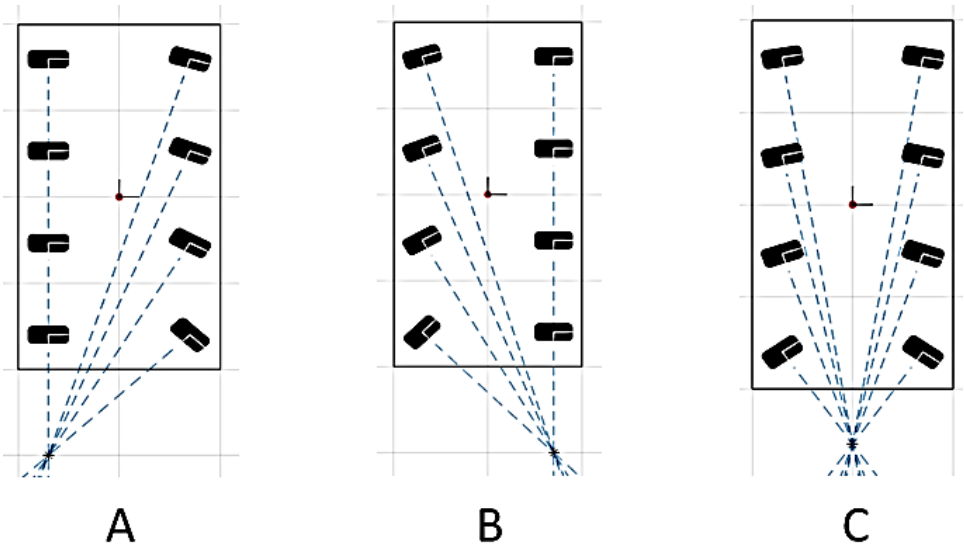

Figure 11. *A: Front wheel steering crosswise, B: Rear wheel steering crosswise, C: Counter-phase steering crosswise.*

The names of the steering programs that are used in this thesis were selected because they were seen to best describe the nature of the different steering programs. The name of the different steering programs can however differ depending on the manufacturer. The rotation mode is for instance called “carousel mode” and crabwise steering is sometimes called “in-phase steering”. To avoid confusion, only the names mentioned in the figures 8-11 will be used from this point forward. Another term related to the behavior of the vehicle movements in addition to the presented steering programs is “omnidirectional steering”. The term “omnidirectional” refers to the ability of existing in all directions. An omnidirectional vehicle is thus a vehicle that is able to move in all directions. Omnidirectional steering is not a steering program, but an attribute that the vehicle has.

Having the option of several steering programs is especially important if the load is large and there is a restricted amount of space to move in. Although the steering programs increase the possibility of handling even larger and heavier loads, just one SPMT is sometimes not enough for handling some loads. It was mentioned earlier that it is possible to link several SPMTs together to form one large SPMT. This may come in handy if the load is large in size and just one SPMT is not enough for handling the load. The connections between the tandem operated SPMTs require sometimes a physical connection and sometimes there is only a data link between the transporters. The leading manufacturers, Goldhofer and Scheuerle to mention a few, seem to support all types of connections and SPMT combinations. Meaning that the computation of turning angles and speeds of the drive units have been done such that the transporters can be connected side-by-side, end-to-end, in a circular form or any other form.

When SPMTs are in tandem, a few things need to be taken into consideration in order for the SPMTs to behave as one single vehicle. One of the things already mentioned are for instance the physical connections and data connections between the SPMTs. The power consumption is another thing. Interconnected SPMTs usually require more than one PPU for handling the possibly large power consumption. The amount of required PPUs depend however a lot on the situation. If the load is large in size but the weight is low, just one PPU may be sufficient enough, but if the load is heavy, it is probably necessary to use several PPUs. The PPU on the most common SPMT types is based on diesel engines that power a hydraulic pump or an electric generator. In indoor environments, a diesel engine is however not the most convenient solution, because diesel engines pollute the air that is supposed to be clean. In indoor environments it is better to use SPMTs that are fully pneumatic or electric instead of the diesel engine based SPMTs. Wheelift[10], Cometto and Solving are companies that manufacture electric SPMTs, sometimes called Electric Modular Transporters (EMTs). SPMTs and EMTs have despite different source of power several things in common, most importantly the computer based drive unit control and the remote controller used for controlling the vehicle.

[10] http://www.wheelift.com/ [Accessed 18.12.2016]

The computer based drive unit control including the vehicle computer, other hardware components and the software is probably one of the most valuable parts of an SPMT. If the drive unit control concept of one SPMT manufacturer would be public, it would not be that difficult to copy the concept and start producing similar SPMTs. The computer based control is a part of the SPMT that require a remarkable amount of engineering, testing and experience, which is why that particular part of the SPMTs is usually kept secret. Although information about the computer based control is difficult to find due to the secrecy, it seems that all manufacturers have different approaches to the drive unit control. Different types of computers or Programmable Logic Controllers (PLCs) are used to implement the control algorithms for the turning angles and speeds of the drive units. According to (Siemens 2014), Siemens PLCs have been used in KAMAG[11] SPMT vehicles. In another article (Novacek 2016) about SPMTs, Wheelift and NASA has together developed a computer based control system using Beckhoff[12] components and especially the new object-oriented programming features in TwinCAT. Scheuerle on the other hand uses its own software called STEPS.3. Information about what type of controllers Scheuerle use could not be found, but a custom made programming software could also mean that Scheuerle use custom made controllers.

In addition to computer based control and PLCs, there are also special types of vehicle controllers intended for these types of applications and these types of applications only. The Kollmorgen CVC600 (Compact Vehicle Controller 600) is a special vehicle controller made for control of different industrial vehicles. Although the CVC600 can be used for control of SPMTs, it is more commonly used to control AGVs. The drive unit control, navigation, communication, logic operations among many other things can be executed on a CVC600. While the construction and layout of wheels on a SPMT or EMT is quite straight forward, AGVs that are built with vehicle controllers such as the CVC600 are a bit more complicated. As mentioned earlier a wheel can be equipped with just a steering motor or just a driving motor or both. In AGVs it is also common to use wheels that are equipped with no motor and can roll and rotate freely around their axes. Such wheels are called swivel wheels or sometimes caster wheels. Fixed wheels are also able to roll freely, but the turning angle is fixed. Other units called diff-wheels consist of two drive-wheels that make the unit rotate by the difference of the velocity of the wheels. The different wheel types used in AGVs are presented in figure 12.

[11] https://www.kamag.com/ [Accessed 18.12.2016]
[12] https://www.beckhoff.fi/ [Accessed 02.10.2017]

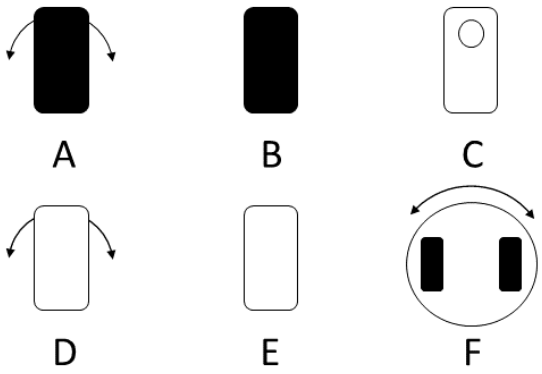

Figure 12. *A: Steer & drive wheel, B: Fixed drive wheel, C: Swivel wheel, D: Steer wheel, E: Fixed wheel, F: Diff-wheel (Solving 2017).*

The SPMTs use steer & drive wheels positioned on the left and right side of the vehicle frame. The distance between each pair of wheels in the vertical direction is constant and the width of each pair is the same for all pairs. The configuration used in the SPMTs is of course also possible to use in AGVs, but the application often require some other type of configuration. The most common AGVs have a steer & drive wheel in the front and two fixed wheels at the rear such that a fork can be positioned between the fixed rear wheels. Kollmorgen calls this type of vehicle an S/D (Steer/Drive) vehicle. An example of additional vehicle types named by Kollmorgen is presented in figure 13.

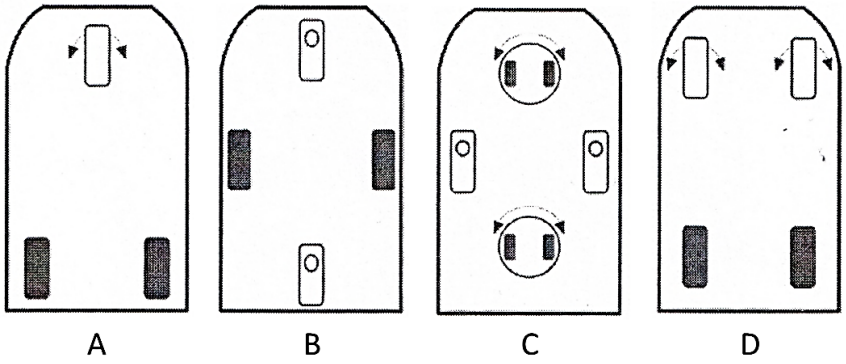

Figure 13. *A: S/D AGV, B: S/D (Diff) AGV, C: Quad AGV, D: Multi-wheeler S/D AGV (Solving 2017).*

Although the vehicle frame can look the same in different vehicle types, there is a significant difference in the behavior of the different vehicle types. Consider for instance the vehicle types S/D and Quad, an S/D type vehicle is not able to drive in the crabwise steering program at all, while that is a quite common steering program for the Quad vehicle.

In addition to the computer based control, AGVs and SPMTs have another thing in common, i.e., the remote controller, which is used for manual operation of the vehicle. Usually it contains two or more joysticks, some buttons, switches and a display. The remote controller can be connected with a wire or wirelessly to the vehicle. The steering programs

can be turned on or off either by the buttons on the remote controller or they can be programmed as specific joystick positions. Other logic operations such as lifting or lowering the load, can also be programmed to be controlled by the remote controller. Note that AGVs do not require remote controllers in normal operation. The remote controllers are only used for occasions when an AGV need to be operated manually. In automatic mode an AGV operates without human interference. The AGV navigates according to a layout with the help of magnetic spots, induction wires, laser scanners or some other technology. Although the AGV gets its reference values from the navigation, the drive unit control is not directly affected by this. The AGV is told where it is supposed to go in the same way in automatic mode as it is in manual mode. The only difference is that in automatic mode the input comes from the navigation and in manual mode the information comes from the remote controller.

2.2.2 Steering Mechanisms in Special Industrial Vehicles

The vehicles presented so far have been equipped with wheels or chains. There are however also other types of industrial vehicles, which operating principle is based on other components than wheels and chains. One special component that is used for moving extremely heavy loads is the air bearing, which is a rubber cushion that can be filled with air. When the air bearing is combined with a metal back plate as presented in figure 14, the combination is commonly called an air caster. A module is usually added on to the air caster, behind the metal back plate to provide for the input holes of compressed-air. The combination of air caster and module for input air is called air bearing module or sometimes air skates or air skids.

Figure 14. *Air caster made by AeroGo[13] consisting of a rubber air bearing mounted on a metal back plate.*

Air bearings are sometimes placed directly under a load or more commonly under a restbar on top of which the load sits. When the air supply is cut off, the restbar sits on the floor and it is impossible to move the load. When the air supply is turned on, the air bearing is inflated and the rest bars as well as the load are lifted off the floor. The air flowing between the air bearing and the floor creates a frictionless thin film of air. Practically this means that everything is floating freely in the air and it is very easy to move the load. The operating principle of an air bearing is further illustrated graphically

[13] http://www.aerogo.com/ [Accessed 19.12.2016]

in figure 15, where the restbars are drawn as a yellow structure between the air bearing and the load.

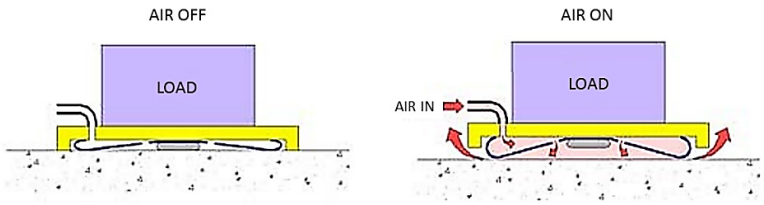

Figure 15. *Operating principle of an air bearing[14]. On the left side is the situation, when air pressure is cut off and on the right side is the situation, when air supply is turned on and the rest bars are lifted off the ground.*

The air bearing modules can be connected to a control unit consisting of a few valves and used as such. For more repetitive tasks, air bearings can be used to build air bearing vehicles. The main components of such a vehicle are the air bearings and drive units as can be seen in figure 16. The vehicle can be fully pneumatic, i.e., even the drive units are built with pneumatic motors or more commonly the drive units are controlled with electric motors, while compressed-air is used only for the air bearings.

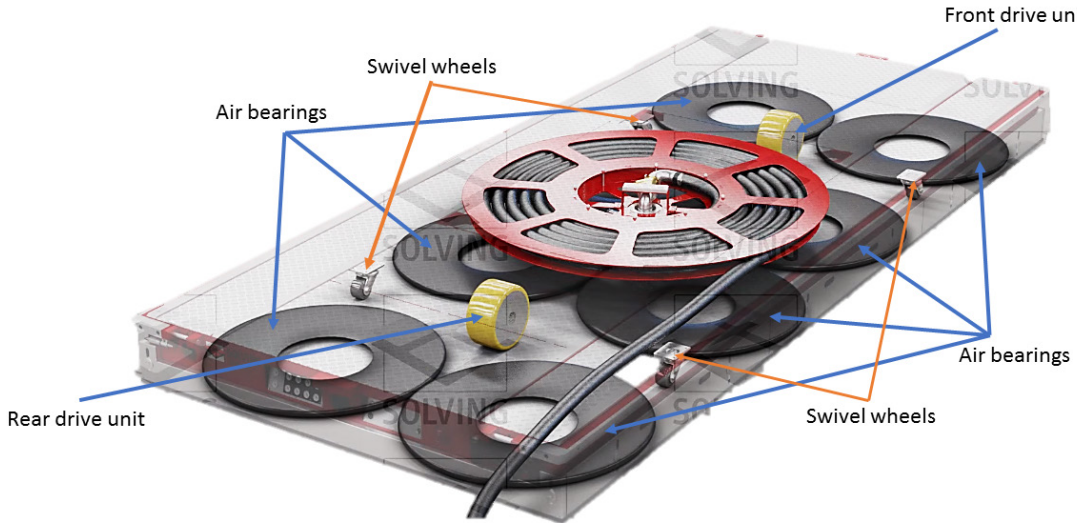

Figure 16. *Main components of an air bearing based vehicle (Solving 2017).*

While an air bearing based vehicle is not lifting anything, the air bearings are empty and the weight of the mover is distributed on the drive units and usually also some swivel wheels. Once the mover is moved under the rest bars and the switch for the air pressure is turned on, the air bearings are inflated. The air bearings will then provide the required lifting force and create a friction-free film of air between the air bearings and the floor. While the air bearings are inflated, the swivel wheels will not touch the floor. The drive units on the other hand are pneumatically suspended, so that although the air bearings are inflated, the drive units will keep touching the floor. The drive units will this way provide

[14] http://www.movetechuk.com/index.html [Accessed 19.12.2016]

for the operability of the vehicle and keeps the vehicle in a fixed position such that it does not start floating freely on the floor. (Solving 2017)

Although the air bearing based vehicles look and operate in a different way than the wheel based vehicles such as the SPMTs, the steering concept and control of the drive units is the same in both vehicle types. The electric drive units are programmed using computer based control and the manual operation is conducted with a remote controller. Just as with SPMTs it is possible to operate several air bearing based vehicles in tandem. In tandem operation there is however for air bearing based vehicles usually no need for more than only two vehicles, operated either side-by-side or end-to-end. The main benefit of air bearing vehicles is that because of the air bearings, it is not necessary to use as many wheels as in traditional SPMTs.

Omnidirectional steering is an interesting feature that enable the vehicle to fit into tight places that would probably not be possible to drive into with conventional steering programs and vehicle components. Omnidirectional steering can be achieved with normal wheels if the range of the turning angle of all wheels is large enough. Another way of achieving omnidirectional driving is to use special wheels designed for omnidirectional driving. These types of special wheels are round as ordinary wheels, but have rolls mounted on the outside perimeter of the wheel. Three of such wheel types that enable omnidirectional driving are presented in figure 17.

Figure 17. *A: The mecanum wheel, B: Omni-wheel, C: WESN (West-East-South-North) wheel (modified from Tadakuma et al. 2007).*

The behavior of the mecanum wheel, omni-wheel and the WESN wheel differ in some minor ways, but the idea is the same. There is no need to turn the wheels, only roll them.

There are also other types of wheels enabling omnidirectional driving than the ones in figure 17. Another wheel type is for instance the omni-ball developed in (Tadakuma *et al.* 2007). Although there are many different types of wheels that could be used for omnidirectional driving, it seems that only the mecanum wheel have been used in applications worth mentioning. Therefore the steering concept of only the mechanum wheel is pursued here. The interesting thing about the mecanum wheel vehicles is that there is no

reason to turn the wheels, because of the special wheels as can be seen in figure 18. The control algorithms for mecanum wheel vehicles are therefore very simple. All omnidirectional movement can be accomplished by controlling only the rolling speeds of each wheel.

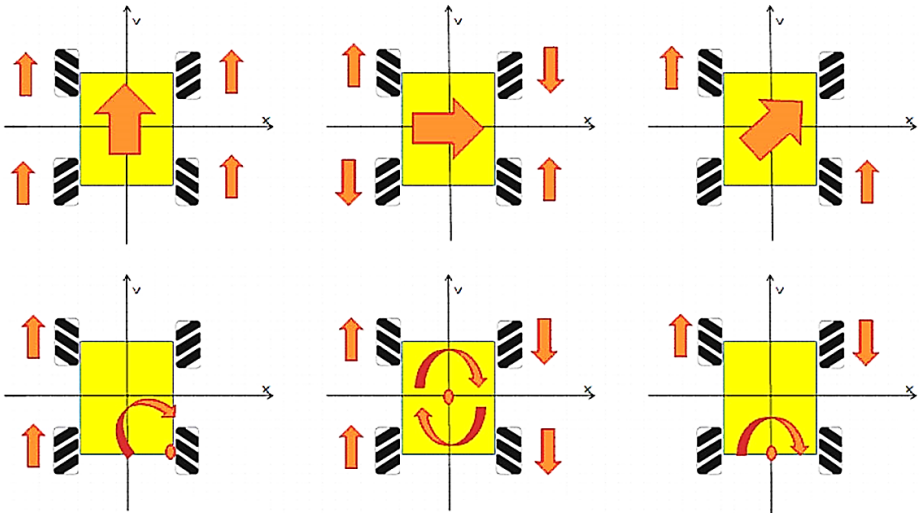

Figure 18. *Operating principle of a mecanum wheel mover*[15].

One of the manufacturers that is currently using the mecanum wheel is KUKA[16]. The KUKA Omnimove UTV (Utility Task Vehicle) is visually quite similar as a conventional SPMT or air bearing vehicle, the characteristics of the movement of the different types of vehicles are however completely different. The difference of the vehicle movements of a KUKA Omnimove UTV and a traditional SPMT is difficult to describe in text, it should be experienced live.

2.3 Solving Vehicles and Requirements on the Proposed Solution

The choice of using wheels, chains, mecanum wheels or air bearings and steering concept overall, depend on the nature of the application. The drive unit based steering is clearly the most flexible steering concept and is applicable for all types of vehicles. In some cases the drive unit based steering concept is however unnecessarily complex. In environments such as construction sites, wheels would most likely get buried in the soft soil, while chains is much better as they divide the weight on a larger surface area and does not get buried in the soil. The choice of steering mechanism can also depend on the vehicle requirements. Same type of features can be achieved with for instance air bearing based vehicles, modular transporters and mecanum wheel vehicles. The environment or load requirements can however restrict the choice of which steering mechanism fits the application best. If the environment is dirty or the floor is uneven, air bearing based vehicles

[15] https://www.roboteq.com/index.php [Accessed 19.12.2016]
[16] https://www.kuka.com/ [Accessed 19.12.2016]

cannot be used. In such a case it would be better to use a mecanum wheel based vehicle or an SPMT instead.

The steering concept that will be developed in this thesis will be developed for Solving's products. Solving produces vehicles that are tailored for heavy load transportation tasks. The main vehicle types Solving offer are custom made AGVs, air bearing based vehicles and electric modular transporters. Solving is specialized in particularly complex and challenging applications. Although each set of Solving products are unique, due to the high level of customization, they can be categorized according to figure 19.

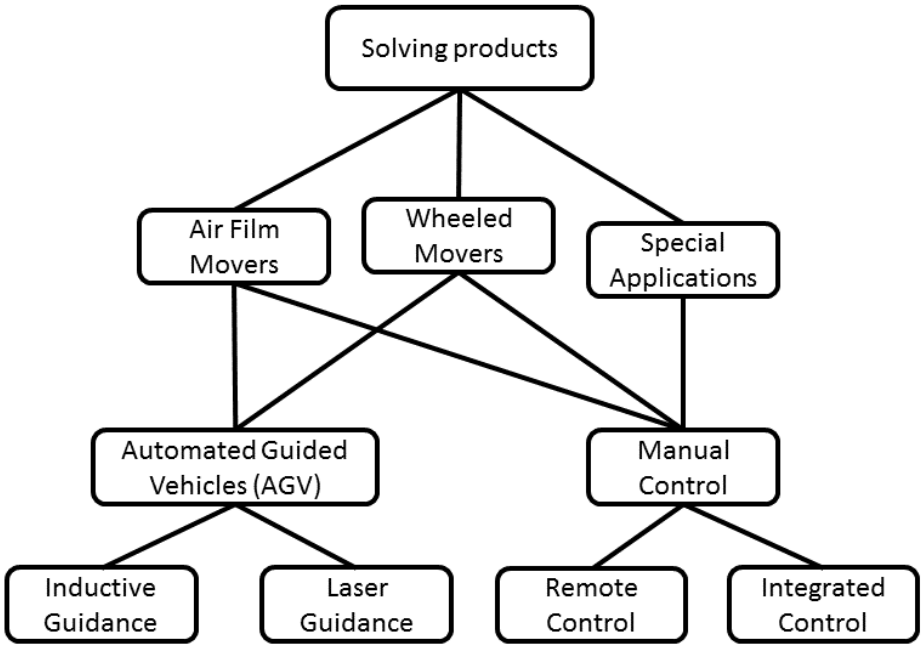

Figure 19. *Solving product portfolio (modified from Sironen 2015).*

The three main product types are: air film movers, wheeled movers and products for special applications. Air film movers are air bearing based vehicles such as the ones that were presented earlier in this chapter. Although the air film movers have been separated from the wheeled movers, they too contain two or four drive units similar to the drive units in the wheeled movers. The drive units in the wheeled movers are not usually suspendable, but they are also pneumatic or electric as the drive units in the air film movers. Air film movers and wheeled movers can be driven individually or in tandem, if such a function is needed. Products made for special applications are not usually wheeled. These types of products move instead on rails for instance. Using air bearings as such for moving heavy loads also go under the "Special Applications"-category.

The three main product types are further subcategorized according to how they are controlled. The vehicle is controlled either manually or it can navigate autonomously, whereupon the vehicle is called an Automated Guided Vehicle (AGV). The AGVs are further subcategorized according to which navigation technology they use. Only the two most common ones have been drawn to figure 19 not to complicate the figure too much, there

are however a lot of other navigation technologies available as well. The most common and precise navigation technology is laser guidance, which is based on a laser scanner located somewhere on top of the AGV. The laser scanner send invisible laser beams around the AGV. The beams are reflected on reflectors that are placed on specific spots across the room the AGV is moving in. The laser scanner measures the distance between the scanner and these reflectors based on the reading from the reflected, incoming beams. The vehicle is able to determine its position by triangulating the measured distances between the reflectors and the laser scanner. (Solving 2017)

The second most common guidance technology is inductive guidance or wire guidance, which is based on an inductive sensor in the AGV and a copper wire buried in the floor. The copper wire is connected to a frequency generator that produces an inductive signal in the copper wire, which the sensor on the AGV is able to detect. Another quite common guidance technology is magnetic spot guidance, which is based on a magnetic sensor in the AGV and magnetic spots or tape located in or on the floor. The operating principle of the magnetic spot guidance is quite similar as the inductive guidance, i.e., a sensor in the AGV follows a line or spots located on or in the floor. Other more uncommon guidance technologies also following a line on the floor include for instance tape guidance, which is sometimes called optical guidance. In tape guidance, the AGV is equipped with an optical sensor that senses a line or spots painted on the floor. The optical guidance was chosen to be called tape guidance in order not to confuse it with the relatively new guidance technology called natural navigation, which is also based on an optical input. In natural navigation a laser scanner on the AGV scans the space, where it is moving and is able to navigate with the help of reference points in the physical space. Although natural navigation is a relatively new technology, it has potential to be one of the more common indoor navigation techniques sometime in the future. In some cases the mentioned navigation technologies have been combined, i.e., some navigation technology is used in one space while another technology is used in another space. (Solving 2017)

Although the air film movers have also been categorized under AGVs, they are more commonly operated manually as opposite to the Solving wheeled movers. Please note that the manually operated Solving wheeled movers are SPMTs or electric modular transporters. The manually operated vehicles including air film movers, wheeled movers and movers for special applications can further be subcategorized in vehicles that are controlled with a remote controller and vehicles that are controlled with a controller integrated on the vehicle. The integrated controller is usually some sort of electromechanical controller located on the vehicle while the remote controller is connected either wirelessly or by wire to the vehicle.

Solving is today using two main types of vehicle computers, i.e., the Kollmorgen CVC600 (Compact Vehicle Controller) and the Siemens ET200SP. The CVC600 allows Solving to use ready-made software and hardware components from Kollmorgen to build complex systems with up to 16 drive units positioned freely within the construction of the frame.

The Siemens controllers are programmed by Solving themselves and used for simple applications with precisely two or four drive units. The drive units need to be moreover positioned in parallel or in-line in relation to one another. Solving would like to increase the use of Siemens controllers in their products, due to several reasons. One of the main reasons is that Kollmorgen components are in general much more expensive than Siemens components and that the availability of and support for Kollmorgen components is more insecure than for Siemens components. Another reason is that the customers sometimes prefer or even require Siemens PLCs, due to Siemens positive status in the field of automation. One thing also to mention is that Solving is currently using and would like to keep using Siemens controllers in all air film movers.

Solving would like to increase the support for amount of drive units in Siemens controlled vehicles. The current challenge Solving is facing is that some air film movers should be built with three drive units in order to support for the weight and size of larger vehicles and loads that customers are currently asking for. The following challenge that Solving will be facing sometime in the near future is to use the same technology for making manually controlled SPMTs. Sometime later on Solving would like to be able to implement the same technology also in AGVs. Solving is obviously not able to do these things with the current program structure. This thesis will present a new flexible program structure made in the Siemens STEP 7 (TIA Portal) programming tool that enable Solving to use an unrestricted amount of freely positioned wheels capable of driving in all steering programs. The features of the Kollmorgen and Siemens controlled vehicles are compared with the new proposed solution in table 1.

Table 1. *Comparison of the current Siemens and Kollmorgen program structures and the proposed solution presented in this thesis ($n \in \mathbb{N}^+$).*

Attribute	Siemens Thesis	Siemens Current	Kollmorgen
Maximum amount of drive units	n	2 or 4	16
Freely positioned drive units	X		X
Changing steering program with a switch	X		X
Changing steering program with joystick directions	X	X	
Support for all steering programs	X	X	X
Changing steering program on the fly	X	X	
Support for manual control	X	X	X
Inductive guidance, tape guidance	X	X	X
Magnetic spot, laser guidance, natural navigation	X		X

The main benefit of the proposed solution compared to the old ones is as can be seen in table 1, the support for n-amount of drive units. While the Kollmorgen based control provide support for up to 16 drive units, which fit Solving's current needs, the support for n-amount of drive units enable Solving to produce and sell vehicles with more than 16

drive units. Considering the manual control, the Kollmorgen based vehicles change steering program using Boolean values that are usually controlled with a physical switch, while the current Siemens based vehicles change steering programs according to Solving's own steering configuration, which is based on the digital outputs of the joysticks on the remote controller. Although it has been discovered that it is possible to override the Kollmorgen program to change steering programs, it is not possible to change the steering program on the fly as it is with the current Siemens based program structure. The new program structure will be made such that it is possible to change the steering configuration, i.e., the way, how the steering programs are changed.

The current alternatives and the new one have three main attributes in common, they support all steering programs, including differential drive, crabwise, front wheel steering, rear wheel steering, counter-phase steering and rotation mode in both lengthwise and crosswise direction. They all also support manual control as well as the basic navigation technologies such as inductive guidance and tape guidance. Although the current Siemens program structure does not support other navigation techniques and it will not be possible to test them with the new program structure, it needs to be done such that it is possible to add support for additional navigation technologies in the future.

In addition to the mentioned functional requirements for the new program structure there are a few non-functional requirements that need to be taken into consideration during the development of the new program structure. The program structure will be used in vehicles handling loads of hundreds of tons, which is why the program structure needs to be thoroughly tested and verified before it is cleared for use in products intended for customers. An error in the code can result in the vehicle turning, accelerating, braking or stopping in an unexpected way. If the vehicle happens to be loaded with a heavy load at the time of an error, there can be dangerous consequences. Unexpected errors such as overflow, underflow or input errors need to be handled in such a way that the vehicle will remain in a safe state until the error has been fixed. Finally the code needs to be structured according to Solving guidelines, flexible to possible hardware component changes, commented and documented such that it is readable and modifiable by employees at Solving even after the completion of this thesis.

3. APPROACH

Systems such as the one in this thesis are called Cyber-Physical Systems (CPS), which is a new, popular term often used in the context of embedded systems and in the field of system design. The word “cyber” stands for the digital computing part of the system and the “physical” part of the term refers to the environment the digital computer is interacting with. A CPS does computations based on inputs coming from the environment, of which the outputs result in a change in the same environment. In this case the input from the environment is coming from a human operator using the remote controller. The digital computation happens inside the remote controller and the PLC, where the values for the drive units, i.e., the turning angle and speed for each wheel is calculated. The human operator sees these changes and turns the joysticks according to where he/she wants the vehicle to go. The same principle applies for an AGV, only that the human and remote controller is replaced with a separate control system.

The program structure made in this thesis will calculate set values for the turning angles and speeds of each wheel in the vehicle. These values are not the final values that will be used as control values for the wheels. The set values that are calculated with the program structure made in this thesis are used as inputs for controllers that are usually P-, PI- or PID-controllers. The outputs from these controllers are finally converted outside the PLC usually in a separate AC (Alternating Current)- or DC (Direct Current)- drive to electrical control signals for the drive and steer motors on the drive units.

The process starting from reading the set values produced by the solution presented in this thesis and ending at the electric output signals calculated by the AC-or DC-drive is vulnerable to disturbances from the environment. The set values must therefore be compared with measured reference values. The reference values in this case, i.e., the true speeds and turning angles of each wheel are measured with sensors such as encoders and potentiometers. The electric signals from the sensors are converted to turning angles and speeds inside the Siemens PLC after which the values can be used for comparison with the set values. The set value calculation provided by this thesis need to function correctly in order for the rest of the system to work. If a set value is incorrect it can result in a wheel skidding, jumping, turning to the wrong direction or in the worst case the vehicle may suddenly turn, accelerate or brake unintentionally.

3.1 Proposed Solution

Each drive unit in the vehicle should be considered unique, i.e., each drive unit has limit values such as maximum speed and turning angle and most importantly a position. The Cartesian coordinate system, i.e., the x/y-coordinate system has been chosen to express

the position of a drive unit. The x- and y-coordinates of a drive unit is easy to derive from the mechanical design drawings of the vehicle or by physically measuring the distances in the direction of the x-axis and the y-axis of the vehicle. Because each drive unit is considered unique, the set value calculation needs to be done separately for each drive unit. In other words there will be n-amount of set value calculations for a vehicle with n-amount of drive units.

The variable inputs that make the wheels turn and roll are the joystick readings. In order for the joystick readings to make any sense, they need to be translated to normalized values located in the same coordinate system as the drive units. While this translation from raw joystick inputs to reference values normalized to the coordinate system could be done in parallel during each set value calculation, the more optimal solution is to separate the translation of the joystick values into a separate translation function. This way the translation can be done only once instead of n-times. The relation between the translation function and the set value calculation functions is presented in figure 20. The translation function from the raw joystick values to reference values is denoted as f_1 and the set value calculation functions are denoted as f_2. In addition to the joystick values, the translation function will need to know some vehicle dimensions such as position of front and rear axis in order to be able to calculate the reference values correctly. Consider for instance how f_1 is supposed to know where the rear axis is when the vehicle is driving in the front wheel steering program. This is need to know so that the turning angle of the wheels on the rear axis can be fixed.

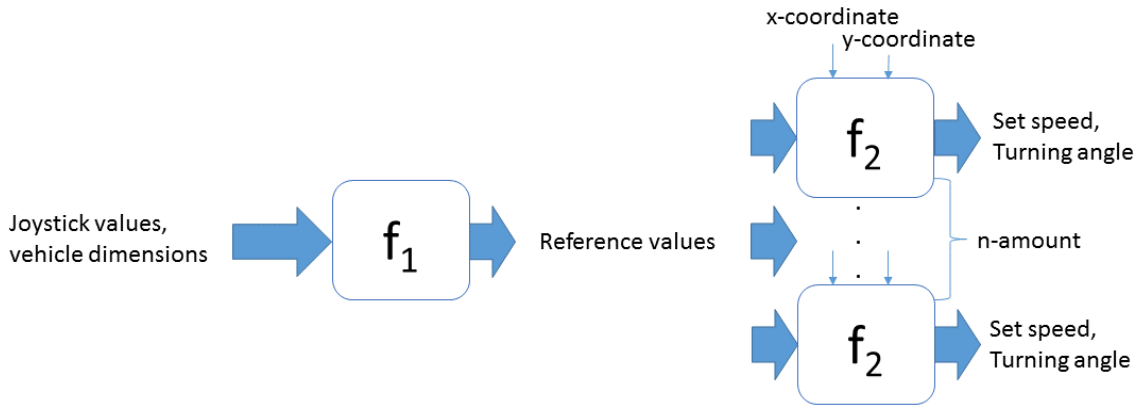

Figure 20. *Set speed and turning angle calculation based on joystick readings, vehicle dimensions and the coordinates of the wheels.*

The reference values mentioned in figure 20 will be explained later in detail, but let it be mentioned at this stage that the reference values consist of a reference velocity, angular velocity, active steering program, heading of the vehicle and the location of the Instantaneous Center of Rotation (ICR). These reference values are used in the functions f_2 to calculate the set speed and turning angle of the wheels.

3.2 Mathematical Analysis

The division of reference value calculation and set value calculation according to the structure presented in the previous section is very optimal also because the heavy calculations can be done only once in the reference value calculation function f_1 and the rest of the calculations can be done in the set value calculation functions f_2. This section is dedicated to theoretical background and deriving the equations and algorithms that are used in the functions f_1 and f_2.

3.2.1 Joystick Readings

The two joysticks that are used for controlling the vehicle are located on the remote controller. Typically the left joystick is used to control the heading of the vehicle and the right joystick is used to control the speed of the vehicle. The output types coming from the joysticks can differ from one another, depending on the remote controller manufacturer, but typically there are at least two analog outputs, one for the analog joystick position in the horizontal direction and another for the analog joystick position in the vertical direction. The analog values can be symmetric, which result in a quadratic joystick space or the analog values can be asymmetric such that the analog values form a circular joystick space. In addition to the analog outputs, most joysticks have four digital outputs, expressing the digital direction of the joystick. A typical setup of the joystick outputs is illustrated in figure 21.

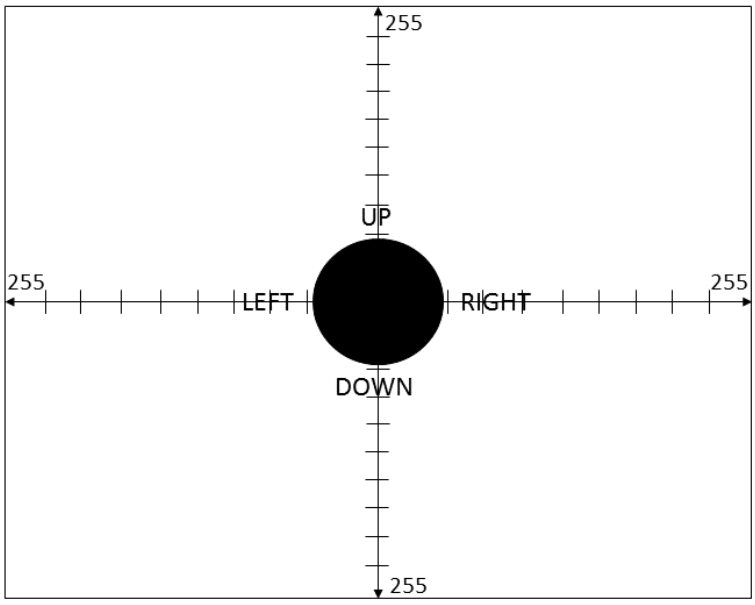

Figure 21. *A typical joystick, which outputs consist of a horizontal and vertical analog value and four digital values. The digital values consist of four Boolean values, UP, RIGHT, DOWN, LEFT. The horizontal and vertical analog values are scaled 0-255 to both directions.*

The joystick in figure 21 has four digital outputs and two analog outputs, which form a quadratic joystick space. The sign of the analog values is the same in all directions, which means that the analog values only express the magnitude of the joystick output, while the digital outputs give the direction of the magnitude. For instance, if both analog values are 255 the joystick may be pointing in any of the four corners in the quadratic joystick space. If on the other hand the digital values UP and RIGHT are true while both analog values are 255, the joystick is positioned in the top right corner of the quadratic joystick space.

One perhaps unusual, but useful quality of the joysticks on the remote controller used during testing of the program structure done in this thesis, was tracks in the digital directions. Deviating from the tracks in the digital directions required some extra force. This could also be noticed in the joystick outputs, while only one of the digital outputs were true as long as the joystick remained in one of the tracks. If the joystick was for instance used in the track of the digital UP direction, the only true digital output was the UP output. This is a quality of the joysticks that has been assumed for all joysticks, especially when using the default Solving steering configuration. If the functions developed in this thesis are to be used with joysticks not having the said quality the digital values may need to be filtered with the help of the analog values. The digital values are a must for the program structure to work.

3.2.2 Detection of Steering Program

The steering configuration, i.e., the way at which the active steering program is determined can be changed in the new program structure developed in this thesis. There are three built-in steering configurations and one example of how a programmer can implement a custom steering configuration. The example steering configuration will not be covered in this document, as the purpose is only to show how fellow software engineers can modify and be creative with the given code. In one of the other steering configurations, the active steering program is changed with Boolean variables that can come from a switch on the remote controller. The remaining two steering configurations that will be covered in this document is the default steering configuration and a modified version of the default steering configuration. The default steering configuration has been developed by Solving and is used in most Solving air film movers today. The modified version of the default steering configuration has been made in order to have an alternative to the default configuration that resembles a configuration of other more familiar vehicles. In both the default and modified steering configurations, the active steering program is determined by the digital joystick outputs.

The direction of the vehicle movement in the default steering configuration is gotten directly from the vertical, digital direction of the right joystick. If the UP value on the right joystick is true, the vehicle will move forward, if the DOWN value is true, the vehicle will move backwards, if neither of them are true the vehicle will remain still. The front wheel steering program is activated when the right joystick is pointing either straight up

or straight down and the left joystick is pointing either left or right. The crabwise steering program is active when both joysticks are pointing in the same horizontal direction and the counter-phase steering program is in contrast to the crabwise steering program active when the joysticks are pointing in the opposite horizontal directions. The rear wheel steering on the other hand is active when the left joystick is idle and only the right joystick is pointing in both a vertical and a horizontal direction. The behavior of a vehicle using the default steering configuration is presented in figure 22. All possible joystick combinations could not fit in figure 22, but the main idea should be clear. On the first row of figure 22 is the direction of movement, on the second row the behavior of the vehicle in default steering configuration and on the third row is the joystick combination used for the steering program presented in the same column.

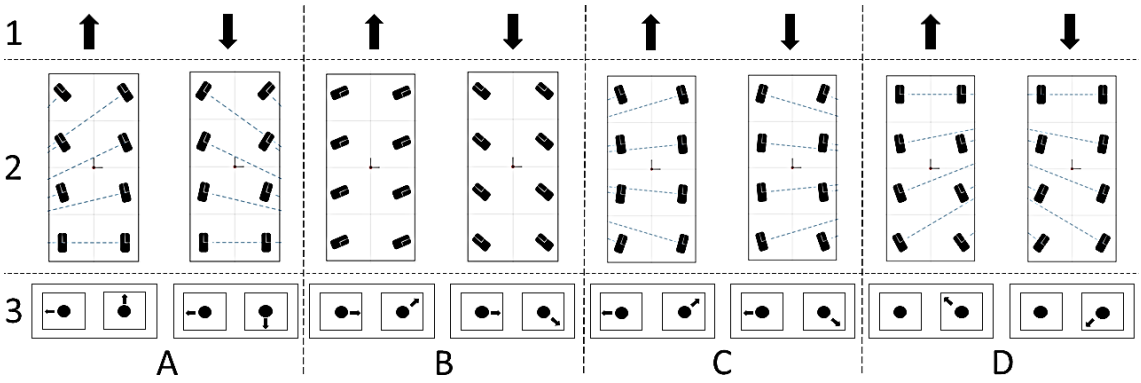

Figure 22. *Row 1: Moving direction, 2: Direction of the wheels on an 8 wheel vehicle, 3: Joystick directions. Colum A: Front wheel steering, B: Crabwise steering, C: Counter-phase steering, D: Rear wheel steering.*

The remaining steering programs that are not presented in figure 22, i.e., differential drive mode and rotation mode are not activated with joystick combinations. They are instead turned on or off with a switch or switches on the remote controller. The speed and direction of the rotation in rotation mode is controlled with the vertical analog value of the right joystick and in differential drive mode the speed of the wheels on the left side is controlled with the left joystick and the wheels on the right side are controlled with the right joystick, similar to skid steered vehicles presented in chapter 2. The differential drive mode and the rotation mode are activated by switches instead of joystick combinations, because lack of joystick combinations, remarkably different vehicle behavior and different interpretation of joystick readings.

In addition to the rotation mode and the differential drive mode, the vehicle orientation can be changed with a switch on the remote controller, assuming it is allowed by the turning angle limits of the wheels. The orientation of the vehicle can be either lengthwise or crosswise. In crosswise mode, the orientation of the vehicle is rotated by +90° in comparison to the lengthwise direction, i.e., the vehicle front will be on the right side of the vehicle, right side is moved to the vehicle rear, rear is moved to the left side of the vehicle and the left side has moved to the front of the vehicle. Otherwise the behavior is the same.

The eight-wheel vehicle presented in figure 22 would in crosswise mode turn only the wheels on the right side during front wheel steering for instance.

3.2.3 Derivation of Vehicle Heading

The vehicle heading is the desired direction of movement of the vehicle. Conventionally, the heading of a vehicle is the direction of the vehicle in reference to some sort of fixed coordinate system. In this case there is no fixed coordinate system in which the vehicle is moving, if there would be it would probably be the factory floor or some other similar environment the vehicle is moving in. The coordinate system used in this case is fixed to the moving vehicle, where the orientation of the vehicle in relation to the coordinate system is always the same and the origin is always the vehicle center. Therefore the vehicle heading in this thesis is defined perhaps a little bit differently than usual.

The desired heading is gotten from the horizontal analog value of the left joystick in all steering programs except rear wheel steering and differential drive mode. In rear wheel steering the desired heading is gotten from the horizontal analog value of the right joystick and in differential mode, the heading is gotten from a combination of the vertical analog values of both joysticks.

The vehicle heading is normalized to a maximum desired heading of the vehicle as follows:

$$\alpha = \frac{max_{\alpha}}{max_{aj}} x_{LJ,RJ}, \tag{1}$$

where α is the desired heading, max_{α} is the maximum value of the vehicle heading, max_{aj} is the maximum of the analog joystick value and $x_{LJ,RJ}$ is the variable, horizontal analog value of the left joystick or right joystick as explained above.

The unit of the desired heading is either degrees or radians and it is supposed to express a change in the orientation. The most intuitive choice would be to normalize the heading to the vehicle center. Expressing the heading through the vehicle center does not however describe the nature of the steering programs. It was additionally noticed after the testing phase that normalizing the heading to the vehicle center in all steering programs caused some challenges in the reference value calculation. The desired heading is therefore normalized to either the vehicle front, vehicle rear or the vehicle center according to figure 23.

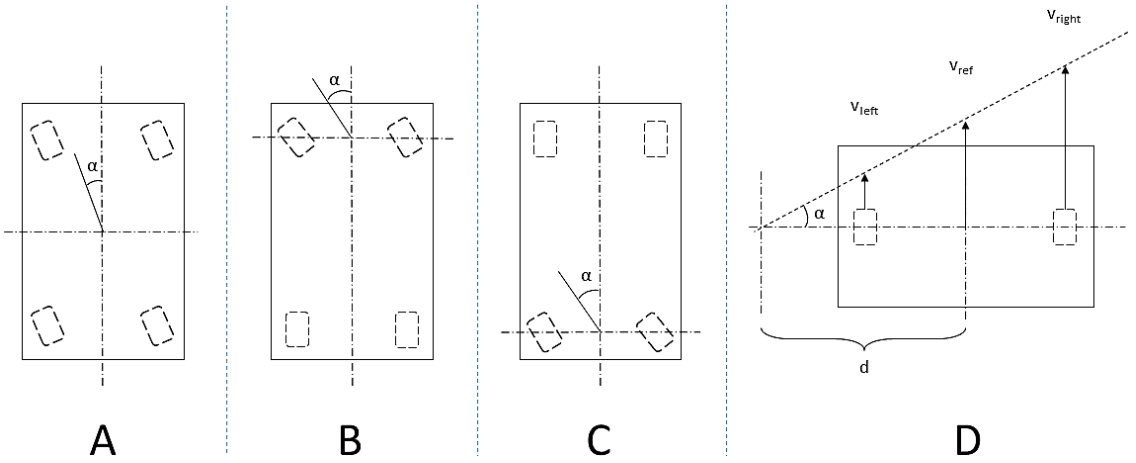

Figure 23. *A: The desired heading normalized to the vehicle center during crabwise steering, B: The desired heading normalized to the vehicle front during front wheel and counter-phase steering, C: The desired heading normalized to the vehicle rear during rear wheel steering, D: The desired heading during differential drive mode.*

There are four different types of headings. One heading expresses the heading of the vehicle center (A), the second of the vehicle front (B), the third of the vehicle rear (C) and the fourth heading is a specific heading used only during differential drive mode (D). In crabwise steering program, the desired heading is normalized to the vehicle center. All wheels are turning the same amount as the desired heading and the speed of all wheels is the same. The heading of the vehicle front, mid and rear are in other words all the same.

In front wheel steering and counter-phase steering, the direction of the vehicle front is changed, which is why the desired heading is normalized to the vehicle front. Following the same principle, the desired heading is normalized to the vehicle rear during rear wheel steering. While the desired heading is the interesting part in the different steering programs, the vehicle will always have a heading of the front, mid and rear, but only one of them are called the desired heading and used in the calculations. Please note that during counter-phase steering the sign of the heading of the vehicle rear is the opposite to the sign of the heading of the vehicle front, hence the name “counter-phase” steering.

In the differential drive mode, the heading is defined in a completely different way than in the other steering programs. In the differential drive mode, the speed of the left side of the vehicle is controlled with the vertical analog value of the left joystick and the speed of the right side of the vehicle with the vertical analog value of the right joystick. An imaginary line intersecting the vectorial joystick values corresponding to the left and right side of the vehicle is created as can be seen in figure 23. The vehicle heading during differential drive mode is furthermore defined as the angle between the imaginary line and the normal line to the wheel planes. The equation of the imaginary line is

$$y = \frac{y_{LJ}-y_{RJ}}{x_{LM}-x_{RM}}x - \frac{y_{LJ}-y_{RJ}}{x_{LM}-x_{RM}}x_{LM} + y_{LJ}, \quad (2)$$

where y_{LJ} is the vertical analog output of the left joystick, x_{LM} is the distance between the vehicle center and the leftmost wheel and x_{RM} is the distance between the vehicle center and the rightmost wheel. By inserting y = 0 into (2), the distance d in figure 23 can be derived as:

$$d = x_{LM} - \frac{x_{LM} - x_{RM}}{y_{LJ} - y_{RJ}} y_{LJ}, \tag{3}$$

where d is the distance between the intersection point of the imaginary line and the normal line to the wheel planes and the vehicle center as shown in figure 23. The distance d and the reference speed, which will be derived later gives the vehicle heading

$$\alpha = \tan^{-1}\left(\frac{v_{ref}}{d}\right), \tag{4}$$

where v_{ref} is the reference speed or the speed of the vehicle center, which will be derived in section 3.2.5. Please note that the sign of all the headings in figure 23 are negative. If the vehicle turns to the opposite direction, the heading will be positive. In differential drive mode it means that the imaginary line intersecting the heads of the speed vectors would be falling instead of rising.

3.2.4 Calculating the Location of the Instantaneous Center of Rotation

The new program structure needs to be dynamic in the sense that it must be able to handle n-amount of drive units. Each instance of the set value calculation functions f_2 must therefore be unaware of each other and there is no use for the vehicle heading alone in the f_2-functions. The functions cannot for instance be aware of where the front axis is during rear wheel steering on the basis of the vehicle heading. Therefore, the Instantaneous Center of Rotation (ICR) is one of the reference values, i.e., inputs to the set value calculation functions f_2. The ICR is the point fixed to the vehicle body that has zero velocity or in other words the point, around which the vehicle creates a circular field. In this thesis it can also be defined as the point, where all normal lines to the wheel planes intersect, as mentioned in chapter 2. In rotation mode it is quite clear that the point fixed to the vehicle with zero velocity, ICR is located exactly at the vehicle center, i.e., the origin (0, 0) in the vehicle coordinate system. In figure 23 it can be seen that the corresponding point in differential drive mode is always located at (d, 0). During crabwise steering on the other hand there is no point fixed to the vehicle body that has zero velocity. For all other steering programs than crabwise steering, the location of the ICR must however be calculated as derived in this section.

In the front, rear and counter-phase steering programs, the ICR follows the path of a straight line. During these steering programs the y-coordinate of the ICR will be fixed to either the vertical position of the rear axis, front axis or the vehicle center. While the y-

coordinate of the ICR is fixed, the quantity that is responsible for steering the vehicle is the x-coordinate of the ICR. Note that the definition of the ICR during front wheel, rear wheel and counter-phase steering programs is that all the normal lines to the wheel planes must always intersect at the ICR as usual when a vehicle is driving according to the Ackerman condition. That is why changing the position of the ICR will cause the wheels to turn.

Because one of the inputs of the reference value calculation function is the vehicle dimensions, i.e., a direct input value and the desired heading was derived earlier, the only unknown in the right triangle formed by the desired heading, the ICR and the vehicle front, mid or rear is the ICR x-coordinate, which can be derived from the tangent of the heading

$$x_{ICR} = \frac{y_{VF,VR} - y_{ICR}}{\tan(\alpha)} + x_{VF,VR}, \tag{5}$$

where $y_{VF,VR}$ is the y-coordinate of either the vehicle front (VF) or vehicle rear (VR) depending on which steering program is active, $x_{VF,VR}$ is the x-coordinate of the corresponding vehicle end and y_{ICR} is the y-coordinate of the ICR. During front wheel and counter-phase steering the, coordinates of the vehicle front, i.e., y_{VF} and x_{VF} are used, while the coordinates of the vehicle rear, i.e., y_{VR} and x_{VR} are used during rear wheel steering.

The equations derived in this section are meant for a vehicle driving in the lengthwise direction. Please note that the same principles apply for a vehicle driving in crosswise direction, only that the coordinate system is rotated by +90°. In other words, the x- and y-coordinates change places. If the vehicle is driving in the front wheel steering program during crosswise mode for instance, the ICR will follow a line perpendicular to the normal line of the leftmost wheel and not the rear wheel as in lengthwise mode.

3.2.5 Calculation of Reference Velocity and Angular Velocity

The derivation of the velocity of the wheels on a vehicle has been inspired by the operational principle of the differential that is a fully mechanical component presented in chapter 2. Meaning that when the vehicle is turning, the velocity of the wheel with the largest velocity will remain constant while the velocity of the other wheels will be scaled down. In practice this means that a wheel's velocity is not restricted only by the joystick readings, but also by the heading of the vehicle. One way to think about it is that the throttle will control the velocity of the wheel or wheels with the largest velocity, while the steering parameter will limit the velocity of the other wheels. The set value calculation functions f_2 will however need to have a reference input in order to be able to calculate the speed for the wheel(s) with the largest speed.

In all steering configurations and steering programs except differential drive mode, the so called reference velocity is derived from the vertical analog value of the right joystick.

The original analog scale of the right joystick is first normalized, i.e., linearly adjusted to the maximum velocity of the wheel that will have the largest velocity, before it is scaled to the velocity of the vehicle center, i.e., the reference velocity. The velocity of the wheel with the fastest speed is in other words calculated as:

$$v_{wh} = \frac{max_{wh}}{max_{aj}} y_{RJ}, \tag{6}$$

where v_{wh} is the normalized value of the vertical analog value of the right joystick and simultaneously the velocity of the wheel furthest from the ICR, $\max_{wh}$ is the maximum allowed wheel and vehicle velocity, $\max_{aj}$ is the maximum value of the analog joystick value and y_{RJ} is the variable, vertical analog value of the right joystick. Because the time it takes for the wheel and the vehicle center to rotate around the ICR is equal, the velocity of the vehicle center is simply

$$v_{ref} = \frac{R_{wh}}{R_{vc}} v_{wh}, \tag{7}$$

where v_{ref} is the velocity of the vehicle center, R_{wh} is the distance between the wheel with the velocity v_{wh} and the ICR and R_{vc} is the distance between the vehicle center and the ICR. The distances to the ICR has been chosen to be denoted with a capital "R" in order to symbolize a radius of the circle being formed around the ICR. The distances denoted "R" are calculated using the Pythagorean Theorem, where the distance between an arbitrarily chosen point (x, y) and the ICR is calculated as

$$R = \sqrt{(x_{ICR} - x)^2 + (y_{ICR} - y)^2}. \tag{8}$$

The normalization of the vertical analog value of the right joystick done in (6) and (7) is a quite straight forward process as long as the vehicle is not rotating. When the vehicle is rotating, the distance between the ICR and the vehicle center is 0, which means that the velocity of the vehicle center is 0. Thus, the velocity of the vehicle center cannot be used as a reference for wheel speed. A better quantity describing rotational movement is angular velocity, i.e.,

$$\omega_{ref} = \frac{v_{wh}}{R_{wh}}, \tag{9}$$

where ω_{ref} is the angular velocity of the vehicle rotating around the ICR. Please note that the velocity of the wheels in rotation mode is of course still controlled by the vertical analog value of the right joystick, hence (6) is used during rotation mode to find the velocity of the wheel furthest from the ICR, after which the velocity is transformed to the angular velocity of the vehicle using (9). Although the angular velocity is a variable only needed during rotation mode it will also be calculated during all other scenarios in case the angular velocity would find a purpose in applications in the future.

Although the reference velocity expresses the velocity of the vehicle center in differential drive mode as in the other steering programs, it is derived in a completely different way than in the other steering programs. In the differential drive mode, the vertical analog value of the right joystick is used to control the velocity of only the right side of the vehicle unlike how it is used in the other steering programs. The reference velocity is derived in an imaginary coordinate system consisting of two vertical vectors with the magnitude corresponding to the vertical analog joystick values of the left and right joystick and some symmetry lines according to figure 24.

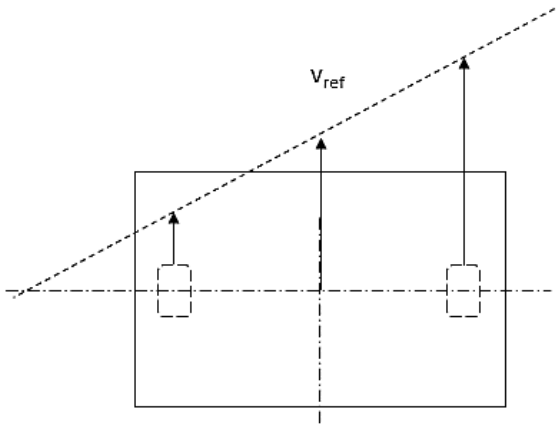

Figure 24. *Normalization of the velocities of the left and right side of the vehicle and the reference velocity during differential drive mode.*

The vector with the same magnitude as the left joystick is placed to the left of the origin at the same horizontal distance from the origin as the leftmost wheel and the other vector is similarly placed at the right side of the origin at the same horizontal distance from the origin as the rightmost wheel. The x-axis of the imaginary coordinate system is the normal line to the wheel planes. An imaginary line is drawn intersecting the heads of both the vectors and the imaginary x-axis. The equation of the line was presented earlier in equation (2). The reference velocity is an imaginary vector located between the left and right vectors at the vehicle center and can be calculated by inserting x = 0 into (2), which gives

$$v_{vc} = -\frac{y_{LJ}-y_{RJ}}{x_{LM}-x_{RM}} + y_{LJ}, \tag{10}$$

where v_{vc} is the velocity of the vehicle center in the imaginary coordinate system. The result of (10) gives in other words a value of the reference velocity normalized to the analog joystick values. To get the final reference velocity, the vector still needs to be rescaled such that it is normalized to a velocity unit. The velocity calculated in (10) is therefore inserted into (6), which gives

$$v_{ref} = \frac{max_{wh}}{max_{aj}} v_{vc}. \tag{11}$$

The reference velocity is in other words first calculated with reference to the imaginary coordinate system before it is finally normalized to the actual velocity of the vehicle center.

3.2.6 Calculation of the Set Speed and Turning angle

Just as the reference values behave differently depending on which steering program is active, the set value calculation behaves differently depending on which steering program is active. When crabwise steering program is active, the speed of all wheels is the same as the value of the reference velocity and the turning angle of all wheels is the current desired heading, i.e., the heading of the vehicle center. Note that there is no need for the coordinates of the wheel in the set value calculation during crabwise mode, because all the wheels are supposed to have the same turning angle and set speed. In all the other steering programs that is not the case.

In differential drive mode all the wheels are locked at their respective idle angle, but the speed of the wheels differ from one another. The set speed for a wheel can be derived from figure 24 and is calculated by

$$v_{set} = (x_{ICR} - x_{wh}) \tan(\alpha), \tag{12}$$

where v_{set} is the set speed of the wheel and x_{wh} is the x-coordinate of the wheel. In the other steering programs, the set speed depends on the distance between the ICR and the wheel. The relative time for a wheel rotating around the ICR should be the same as the relative time for the vehicle center rotating around the ICR, which gives

$$\frac{2\pi R_{wh}}{v_{set}} = \frac{2\pi R_{vc}}{v_{ref}}. \tag{13}$$

Because the distances R_{wh}, R_{vc} and v_{ref} can be calculated using the equations derived earlier, the only unknown in (13) is the set speed, which can be solved as

$$v_{set} = \frac{R_{wh}}{R_{vc}} v_{ref}. \tag{14}$$

The set speed calculation is quite a straight forward process. The turning angles during front wheel, rear wheel, counter-phase steering and rotation mode on the other hand are a bit more difficult to process. There are several approaches of calculating the turning angles following the Ackerman condition that are presented in (Bishikar 2014), (Giancarlo et. al 2009) and (Jazaar 2008) among others that handle the calculation of turning angles. Each approach could be extended to multi-wheeled vehicles, but they all assume that the relationship between the wheels is somehow fixed, which cannot be assumed in this case, where it must be possible to position a wheel freely in the Cartesian coordinate system. The approaches could be used when creating a vehicle with a specific layout of the wheels, but not in this case.

The turning angle of a wheel, is the angle between the wheel plane and the positive y-axis of the vehicle. The turning angle is negative when the wheel is turning counterclockwise, i.e., left and positive when the wheel is turning clockwise, i.e., right just as the sign of the heading. A wheel with the turning angle θ is presented in figure 25.

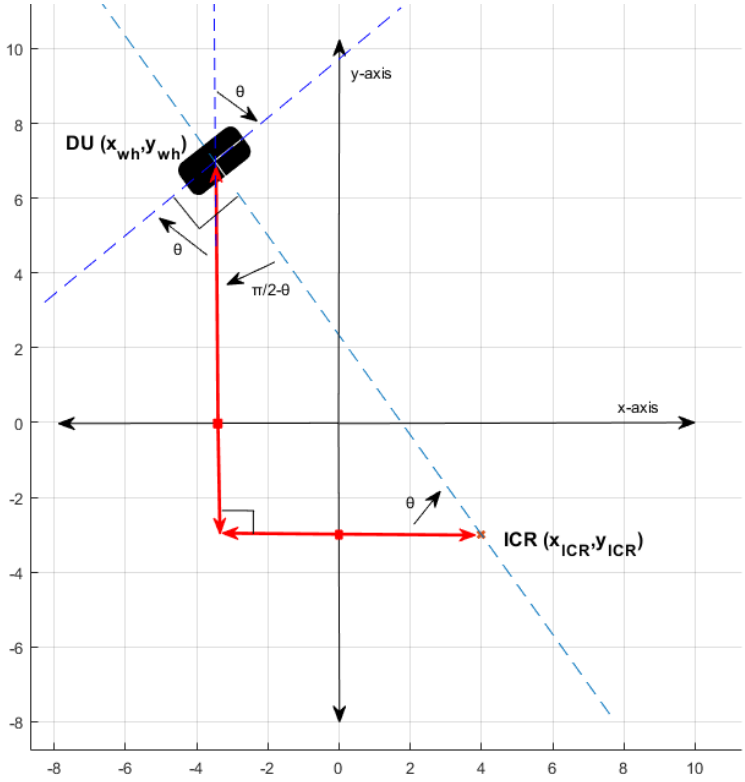

Figure 25. *Freely positioned wheel with turning angle θ. The coordinates of the wheel is (x_{wh}, y_{wh}) and the coordinates of the ICR that the normal line to the wheel plane is intersecting is (x_{ICR}, y_{ICR}).*

Figure 25 has been drawn following the fact that the turning angle θ is the angle between the wheel plane and the positive y-axis of the vehicle and that the normal line to the wheel plane intersect with the ICR. The unsigned value of the turning angle is

$$\theta = \tan^{-1}\left(\frac{\|\overrightarrow{y_{wh}} - \overrightarrow{y_{ICR}}\|}{\|\overrightarrow{x_{ICR}} - \overrightarrow{x_{wh}}\|}\right). \tag{15}$$

Considering that the sign of the turning angle is supposed to be positive when the wheel is turning right and negative when the wheel is turning left, the signed turning angle is

$$\theta = \tan^{-1}\left(\frac{y_{wh} - y_{ICR}}{x_{ICR} - x_{wh}}\right), \tag{16}$$

which is the signed turning angle of any wheel during the front wheel, rear wheel, counter-phase and rotation mode steering programs.

3.2.7 Exceptions that may Occur During the Calculations

In the beginning of this chapter, all the reference values were mentioned, but no motivation or explanation for them were given yet at that stage. It has been shown that the reference values all behave somewhat differently depending on what steering program is active. The purpose of each reference value was shown by mathematical derivation beginning from the inputs such as the digital and analog joystick outputs and ending at the final set speeds and turning angles of the wheels. Several exceptional situations may occur during this calculation process. One of the most significant exceptions occur when a heading is outside the interval]-90°, 90°[. Consider the situation in figure 25 for instance. How is the set value calculation function f_2 supposed to know if the turning angle is indeed θ or θ - 180°? By comparing the signs of the calculated turning angle and the desired heading, the set value calculation function can compensate for possible errors. In other words, the set value calculation function will add or subtract 180° from the calculated turning angle, when necessary.

Dangerous exceptions may occur if some of the set speeds or turning angles get an undefined value such as overflow, underflow or NaN (Not-a-Number). The speed or turning angle may then suddenly jump from a very low value to a high invalid value and if the vehicle is poorly designed it can cause the wiring in a drive or steer motor to overheat, which in worst case can lead to a fire. That is why all outputs, also the reference values need to be checked for situations such as overflow, underflow and NaN, even though they are most likely related to faulty inputs. If such an error occur, the best solution is to put the vehicle in idle-mode with all turning angles and speeds set to zero.

Other exceptions related to faulty inputs occur for instance when the vehicle width is zero and the vehicle is trying to activate differential drive mode, which is clearly not possible when the vehicle width is zero. Because all wheels are vertically in-line, it’s impossible to steer as supposed to during differential drive mode, i.e., the speed of the left side with the left joystick and the speed of the right side with the right joystick. In this particular situation the steering program will be forced to change to crabwise steering instead of differential drive, while all the turning angles of the wheels remain idle.

Handling exceptions is one of the parts of the program structure that lifts the level of quality of the code and is an essential part of the program structure developed in this thesis. All specific exceptions can however not be handled in this document, due to several reasons, of which most importantly lack of space. The examples mentioned in this section should however give an idea of what types of exceptions are checked in the code. The procedure done when an exception happens is explained in more detail in the next chapter.

3.2.8 Summary of Mathematical analysis

Because the active steering program is such an essential part of the program, it must be determined first before starting actual calculations. The active steering program is determined either with a switch, as done in one of the steering configurations or with a look-up-table (LUT) made of the digital outputs (DO) of the joysticks. After determination of the active steering program, the reference value calculations are done in the reference value calculation function that was denoted as f_1. The reference values consisting of the active steering program, vehicle headings, reference velocity, angular velocity and the x- and y-coordinates of the ICR are used as inputs in all of the n-amount of set value calculation functions that were denoted as f_2. The set value calculation function calculates the final set values of speed and turning angle for one wheel.

4. IMPLEMENTATION

The outcome of this thesis will affect everyone at Solving all the way from the salesman to the final customer. The salesman will be able to sell Siemens-based vehicles with more wheels than was possible before this thesis, which on the other hand leads to several benefits from the customer's perspective. A vehicle with a lot of wheels will divide the load on a larger surface area and creates a smaller pressure on the floor or ground it is moving on than a vehicle with less wheels. Solving will be able to produce electric modular transporters based on Siemens PLCs, which they were not able to produce before. The Siemens based electric modular transporters may even interest new customers and create business opportunities for Solving that were not possible before this thesis. The mechanical and electrical engineers at Solving will have to design for these types of vehicles and the production workers will have new types of products to assemble. Most importantly the software engineers at Solving will have to learn to use a new program structure.

The most important stakeholders have been taken into consideration during the entire development process of the new program structure. The physical and virtual parts of the vehicle connected to these stakeholders can be divided into several types of interfaces. The stakeholders, interfaces and a few of the most important use cases are presented in figure 26. The use cases are presented using the UML (Unified Modelling Language) use case diagram. UML will be used in several of the figures in this chapter to express the choices made in the final implementation of the new program structure.

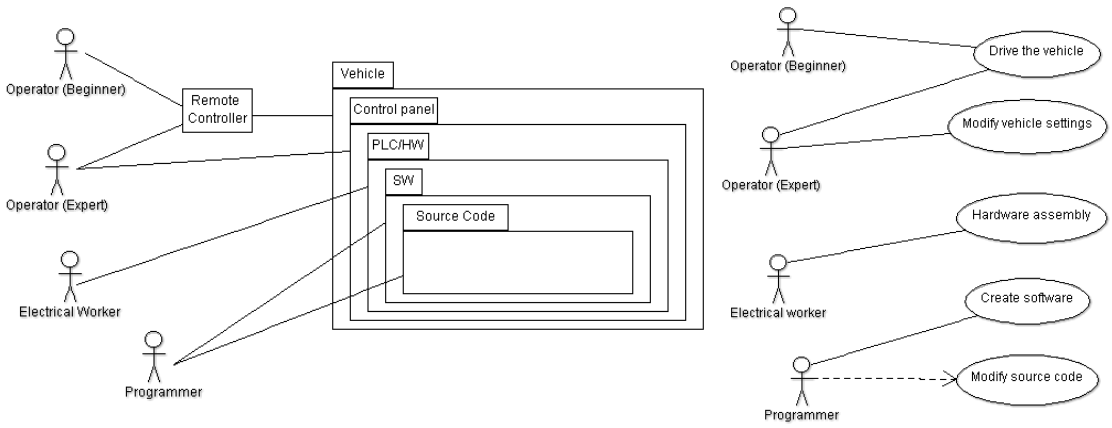

Figure 26. *Vehicle interfaces and a UML use case diagram of the most important use cases.*

The customers that will be operating the vehicle have been divided into two main types of users, beginner level operators and expert level operators. The beginner level operators drive the vehicle using the remote controller and use only the most basic steering programs such as front wheel and rear wheel steering. The expert level operator on the other

hand is comfortable using all the steering programs. The expert level operator can additionally modify vehicle settings using the web-interface or a control panel that is usually located on the remote controller.

The electrical worker assembles the hardware and sees to it that the electrical connections in the vehicle are done correctly. The electrical connections are the reason why a signal from the PLC can cause the wheels to turn and roll. The software engineer on the other hand makes sure that the electrical signals behave correctly. The software created by the software engineer, sees to it that the signals coming from the remote controller result in the desired behavior of the vehicle. The most basic vehicles can be programmed by fine-tuning a few parameters or making changes in the vehicle software on the functional level, i.e., usually using the FBD (Function Block Diagram) programming language in the Siemens programming environment, while in more rare cases, the software engineer needs to modify the software on a more detailed level.

From the perspective of the program structure developed in this thesis, each stakeholder must follow their responsibilities, if the program structure is to function properly. The software engineer that will be in direct contact with the program structure must connect the right inputs and outputs to the program blocks, the electrical worker must see to it that all hardware is used as planned, that the encoders and potentiometers are installed in the correct orientation and position for instance. If an encoder is installed wrong, the corresponding wheel can turn to the wrong direction. The expert level operator must use logical vehicle settings defined in the web-interface. Because if the operator changes a setting like the vehicle width, the differential drive mode cannot be activated in some cases for instance.

The beginner level operator must be able to use the remote controller to drive the vehicle and change steering programs either via the digital outputs of the joysticks or a switch on the remote controller. The expert level operator should additionally, depending on what rights the operator is given, be able to enable or disable steering programs, change steering configuration, change attributes of the drive units such as maximum allowed speeds and turning angle limits and change the vehicle dimensions.

The program structure must be done such that it does not create extra work for the electrical worker. Poor software design can sometimes result in more work on the hardware. Most importantly the software engineers that will be using the new program structure must be able to do their work. The names of the inputs and outputs used in the program structure must be well deliberated, the communication flow needs to be logical, the functions and blocks must be documented in detail and most importantly the source code needs to be made such that if a software engineer unfamiliar with the program structure needs to modify the source code it will be intuitive to get into. It is possible that a customer is for instance asking for a completely custom made steering configuration. In that case,

the software engineer needs to be able to learn about the program structure by reading the documentation and implement the steering configuration in the source code.

4.1 Block Diagram of the New Program Structure

The program structure is implemented in the Siemens STEP 7 (TIA Portal) programming environment using Siemen's text-based programming language called SCL (Structured Control Language). The language is related to the IEC 61131-3 standardized ST (Structured Text) language, but compared to ST there are a few more features in SCL. The interface of the reference value calculation and set value calculation functions presented in chapter 3, have been done in the FBD (Function Block Diagram) language. The created program structure consists in other words of two types of FBs (Function Blocks), one reference value calculation FB and one set value calculation FB. The SCL is thus used for the internal implementation of the FBs. The idea is that only one of the reference value calculation FBs and n-amount of the set value calculation FBs is to be used in vehicle programs (n is the amount of wheels as explained before). Function Blocks as well as the SCL programming language are familiar to all software engineers at Solving, which justifies the choice of using FBD and SCL.

FBs are intuitive to use and a natural choice for functions such as the ones made in this thesis. Only the inputs and outputs are visible to the programmer and there is no need to study the internal implementation of the functions. If however seen necessary, for instance when a customer require a custom made steering configuration, it is no problem for the programmer to edit the internal implementation of the FBs. It is also possible to protect function blocks, which could be useful if the customer is allowed access to the software of the vehicle, but the internal implementation of the FBs is desired to be kept secret. A block diagram of the program structure is presented in figure 27.

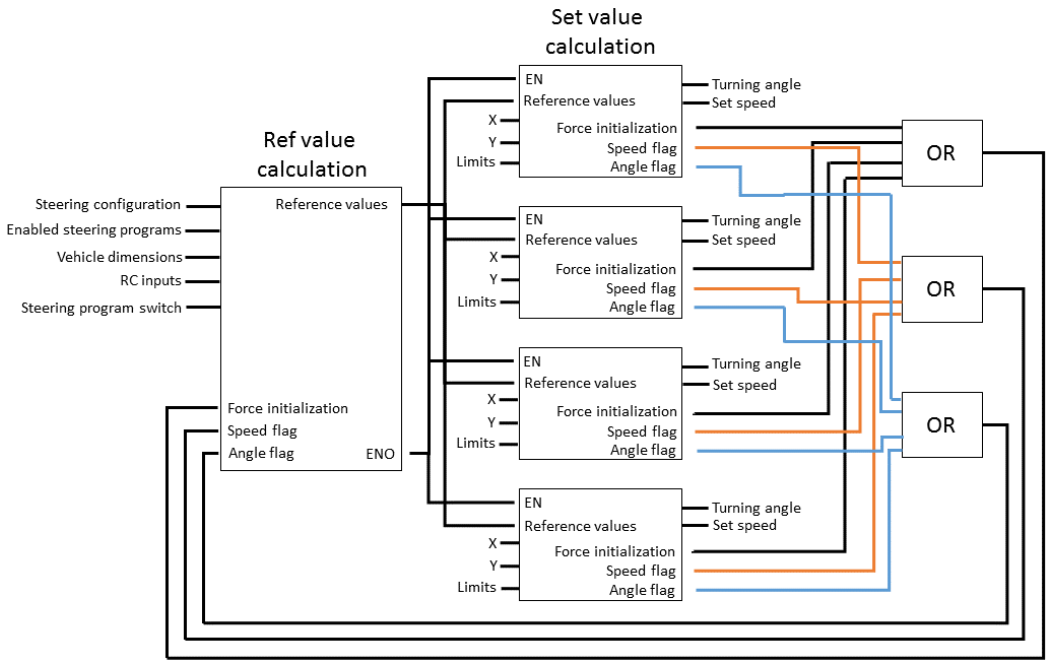

Figure 27. *Block diagram of the new program structure.*

Because four of the set value calculation functions are used, it can be derived that the program structure of figure 27 is being used in a vehicle with four wheels. The inputs of the reference value calculation function block are the steering configuration, enabled steering programs, the vehicle dimensions, i.e., position of the front and rear axis in both the lengthwise and crosswise direction, the remote controller signals such as the joystick readings and switches for the differential drive mode, crosswise/lengthwise switch and rotation mode switch. The last input of the reference value calculation FB determined by the programmer is meant for an optional switch for activating or deactivating steering programs that is used in the steering configuration, where steering programs are activated and deactivated with a switch.

The main purpose of the reference value calculation function is to use the presented inputs to calculate the reference values that will be used in the set value calculation functions. The reference values are as mentioned earlier the active steering program, reference speed, angular velocity of the vehicle, the x- and y-coordinates of the ICR and the vehicle headings. The EN and ENO signals drawn in figure 27 are characteristic of the Siemens PLC programming languages. When the EN signal is true, the FB is enabled and vice versa. The ENO signal is usually used to enable subsequent FBs, subordinate to the FB outputting the ENO signal. In figure 27, the ENO signal of the reference value calculation FB is used to enable the set value calculation FBs. This way the ENO coming from the reference value calculation function can be used to disable the set value calculation functions in case of a serious error.

Each set value calculation FB has a unique set of inputs consisting of the x- and y-coordinates of a wheel and the limiting values, i.e., drive unit type, maximum speed and maximum positive and negative turning angles. The interesting results of the program structure are the outputs of the set value calculation FBs, i.e., the turning angles and set speeds of each wheel. Those outputs will be used in other parts of the program before they are finally converted to electrical signals for the drive and steer motors of the drive units.

To keep the final set values within a reasonable range, the reference values must be scaled and restricted accordingly. Safe values, i.e., values when the turning angles and set speeds do not exceed their respective maximum values, are calculated as soon as the FBs are enabled during a process that is called the initialization process. After the process has finished, the reference values are scaled according to the safe values, so that they will never cause a turning angle or set speed to exceed their maximum limits. If some attribute of a drive unit changes, for instance if the maximum speed limit of a drive unit changes, the initialization process needs to be run again. The “force initialization” flags of the set value calculation FBs are therefore connected to the corresponding input of the reference value calculation FB. This way the set value calculation FB can inform the reference value calculation FB, if a new run of the initialization process needs to be done. Please note that when using more than one drive unit as in figure 27, the flags need to be logically ORed in order to account for the flags of all drive units. The initialization process will be explained in detail later.

In addition to the “force initialization” flag there are two other types of flags, i.e., the angle flag and the speed flag. They are used to express if the corresponding value has exceeded a limit, i.e., if the set speed goes over the maximum speed limit, the speed flag will be true and correspondingly if the turning angle goes over the maximum turning angle limit, the angle flag will be true. The speed flag and angle flag are mostly used during the initialization process, but are kept in use during normal operation as well. During the initialization process, the angle and speed flags are used to signal the moment when either the turning angle or set speed goes over a limit value. During normal operation, the angle and speed flags are used as a safety precaution, meaning that if for some reason the initialization process has failed or the joystick values have been incorrectly scaled such that the maximum set values can be exceeded, the flags will signal the reference value calculation FB to stop changing the reference values, so that the set values do not exceed the limits.

Although the program structure in figure 27 has been drawn for four drive units, the same principles applies for a vehicle with n-amount of drive units. If a drive unit were to be added to the vehicle with the program structure in figure 27, a set value calculation FB would be added with its inputs and outputs connected similarly as for the currently presented set value calculation functions. The logical OR functions would of course need an additional input gate for the flag signals from the added set value calculation FB. The structure of figure 27 does therefore support the original requirement of being able to

handle an unrestricted amount of drive units. The choice of using the FBD language for the interface of the functions and SCL for implementing the internal functionality of the FBs is a choice that made the work of this thesis easier and will help the software engineers at Solving to better understand the program structure. The reference value calculation FB and the set value calculation FB will be explained in detail, in the following sections.

4.2 Structure of the Reference Value Calculation FB

The equations presented in chapter 3 do not as such result in a significant amount of code, but non-functional requirements has forced the lines of code (LOC) to be quite large. The inputs must be checked for errors before starting the calculations for instance, the FB must be able to detect errors during the calculations and the code is written in a style that is intuitive for other programmers to read. The reference value calculation FB is presented in the form of an activity diagram in figure 28.

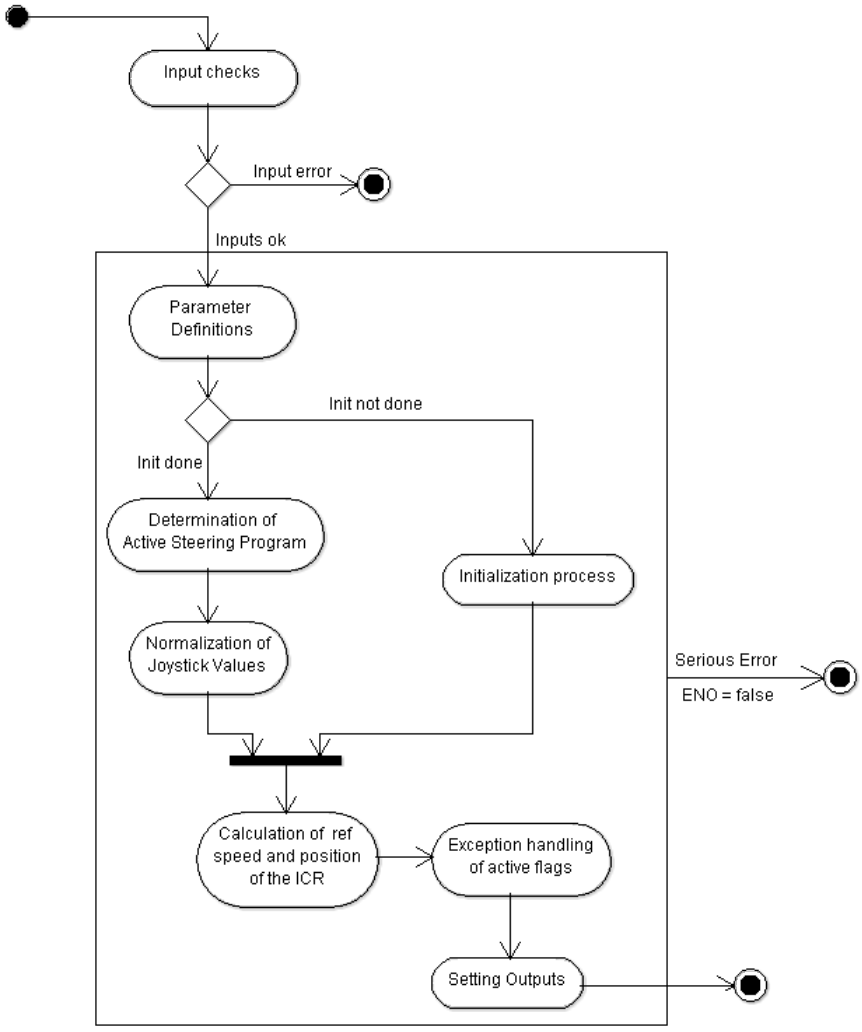

Figure 28. *UML activity diagram of the reference value calculation FB.*

At the beginning of every execution cycle of the reference value calculation FB presented in figure 28, the inputs are checked so that they are valid. If the input signals are ok, the program continues by defining a few helper parameters that will be used later in the FB in the calculations. Assuming the initialization process has finished, the next step is to determine the active steering program and normalize the joystick values. If the initialization process has not finished, the FB will fabricate an active steering program and normalized joystick values. At the end of the FB, before setting the outputs, the x- and y-coordinates of the ICR as well as the reference velocity and angular velocity are calculated. In case of a high flag, the reference values are fixed at safe values until the reference values change, so that it results in a smaller turning angle or set speed of the wheels.

There are three ways to end the execution of the FB. If there is an error in the inputs or a serious error during the rest of the execution, it is terminated immediately. When the execution is terminated like this, the vehicle is set in idle mode with all turning angles and set speeds at zero and the ENO output is set false as explained earlier. If there are no errors in the inputs nor during the execution, the execution is ended normally, which is the third way to end the execution. The outputs are then set to the values, calculated during the execution.

If the programmer needs to change the SCL code of the FB, to implement a custom steering configuration for instance, the only parts of the code that should need editing is the determination of the active steering program and the normalization of the joystick values. Although the normalization of the joystick values for the built-in steering configurations is quite similar for all steering configurations, the possibility of other normalized values for other steering configurations have been made possible. The SCL code has been designed such that the programmer can easily find where to insert the code for custom steering configurations. In the following subsections each part of the reference value calculation FB, presented in figure 28 are explained.

4.2.1 Input Checks

The purpose of the input checks is to terminate the execution of the FB immediately, if there is an error in the inputs. If both the digital outputs LEFT and RIGHT are true on one of the joysticks for instance, it will be impossible to determine the active steering program. The execution must therefore be terminated immediately at that moment. Another important check that is done is the vehicle dimensions. The distance between the vehicle center and the front and rear axis in both lengthwise and crosswise directions are for instance checked so that they are less than 1 km, i.e., a value that would result in an unrealistically large vehicle.

4.2.2 Parameter Definitions

In the beginning of the programming phase of this thesis, it was noticed that the calculations required parameters that were to be used at several different stages in the code. These types of parameters are therefore calculated at the beginning of the FB, instead of always calculating them at the place, where they are used. These types of parameters are the position of the vehicle front for instance, which by default is at the front of the vehicle, but in crosswise mode it is located at the right side of the vehicle.

One particular parameter that was found especially useful, because the digital directions of a joystick were found difficult to work with in some situations, is the cardinal direction of a joystick. The cardinal directions are also safer to work with, because one cardinal direction include all the digital directions. When a joystick is pointing to the upper right corner for instance, the cardinal direction is northeast (NE), which means that all four digital joystick outputs have been checked such that the RIGHT and UP values are true and the LEFT and DOWN values are false. If the digital values would be used, the NE direction can be checked by checking only the UP and RIGHT values of the digital directions, which means that there may be an undetected error if for instance the LEFT or DOWN value is true as well. The translation from the digital joystick readings to cardinal points is presented in table 2,

Table 2. *Conversion from digital joystick readings to cardinal direction.*

Cardinal point	UP	RIGHT	DOWN	LEFT
N	TRUE	FALSE	FALSE	FALSE
NE	TRUE	TRUE	FALSE	FALSE
E	FALSE	TRUE	FALSE	FALSE
SE	FALSE	TRUE	TRUE	FALSE
S	FALSE	FALSE	TRUE	FALSE
SW	FALSE	FALSE	TRUE	TRUE
W	FALSE	FALSE	FALSE	TRUE
NW	TRUE	FALSE	FALSE	TRUE

where each cardinal direction, i.e., north (N), northeast (NE), east (E), southeast (SE), south (S), southwest (SW), west (W) and northwest (NW) should be familiar from other context. The cardinal directions will not be further illustrated here, because they should be quite a familiar concept to most people. The cardinal points made the next step of the FB much easier and reduces the amount of LOC significantly.

4.2.3 Determination of Active Steering Program

Steering programs are activated or deactivated according to a look-up-table (LUT). The LUT is built differently depending on which steering configuration is used. In the steering

configuration, where the steering programs are activated or deactivated with a switch, each steering program has a corresponding Boolean value. If the Boolean value is true, the corresponding steering program is set active. Note that only one Boolean value at a time can be true. In that case the LUT is quite simple, each Boolean value results in the corresponding steering program getting active. In the rest of the steering programs the LUT is built of the cardinal points of the joysticks. The LUT of the default steering configuration is presented in table 3.

Table 3. *Illustration of how the direction of the vehicle movement and steering program is determined based on the cardinal directions of the joysticks in the default steering configuration.*

Left joystick	**Right joystick**	**Direction of vehicle**	**Steering program**
IDLE, N, S	IDLE	NONE	IDLE
IDLE, N, S	N	forward	Crabwise
N, NW, W, SW, S	W	NONE	Crabwise
N, NE, E, SE, S	E	NONE	Crabwise
N, NW, W, SW, S	NW	forward	Crabwise
N, NE, E, SE, S	NE	forward	Crabwise
IDLE, N, S	S	backward	Crabwise
N, NW, W, SW, S	SW	backward	Crabwise
N, NE, E, SE, S	SE	backward	Crabwise
ALL except N,S	IDLE	NONE	Front wheel steering
ALL except N,S	N	forward	Front wheel steering
ALL except N,S	S	backward	Front wheel steering
IDLE	W, NW, N, NE, E	forward	Rear wheel steering
IDLE	SW, S, SE	backward	Rear wheel steering
NW, W, SW	E	NONE	Counter-phase
NE, E, SE	W	NONE	Counter-phase
NW, W, SW	NE	forward	Counter-phase
NE, E, SE	NW	forward	Counter-phase
NW, W, SW	SE	backward	Counter-phase
NE, E, SE	SW	backward	Counter-phase
Rotation mode is turned on with a switch			
-	S	CW	Rotation
-	N	CCW	Rotation
Differential drive mode is turned on with another switch			

According to table 3, rotation mode and differential drive mode are indeed turned on or off with switches as stated in chapter 3. The conversion from the joystick outputs to active steering program has also been specified in more detail as in chapter 3. If the right joystick is idle for instance and the left joystick is pointing either left or right, the front wheel steering program is activated by default and a steering program called “IDLE” has been added to express the state when all the turning angles and set speeds are zero.

In any programming language a table can be converted to a bunch of if-statements, which has naturally been done also in this thesis. Each if-statement in the code correspond to a row in the table. The rotation mode and differential drive mode are handled first, because they are controlled with switches. Otherwise the if-statements follow the order of the table. Note that the active steering program is need to know for calculating the position of the ICR and reference velocity as was shown in chapter 3.

4.2.4 Normalization of Analog Joystick Readings

The normalization of the analog joystick readings is a simple process of linear adjustment. In this case it is the analog joystick values that are converted to a desired heading and speed according to the equations in chapter 3. Even though the normalization stage is quite similar in all the built-in steering configurations, an open “CASE”-structure has been made to keep the possibility of making custom normalizations.

In the default steering configuration, the desired speed in all steering programs except differential drive mode is gotten from the vertical analog value of the right joystick. In the differential drive mode, the desired speed is instead gotten from a combination of the vertical analog values of both joysticks. The desired heading on the other hand, is gotten from the horizontal analog value of the left or right joystick. The horizontal analog value of the right joystick is used in rear wheel steering and the horizontal analog value of the left joystick in the other steering programs.

In the steering configuration that uses switches for changing steering programs, the horizontal analog value of the left joystick is always used for the desired heading and the vertical analog value of the right joystick is always used for the desired speed. Although the built-in steering configurations use the vertical and horizontal analog values in the normalization, the “CASE”-structure enable the software engineer using the FBs to use whatever value for calculating the desired speed and heading. The phase or magnitude of the analog joystick values could for instance be used instead of the direct analog values.

4.2.5 Initialization Process

The purpose of the initialization process is to find the normalization constants or more specifically the max_{α} and max_{ref} values that are used in the normalization step. The initialization process runs every time the reference value calculation FB is enabled or when the initialization is forced by the “force init”-input of the reference value calculation FB. During the initialization process, the turning angles and set speeds are modified only virtually, i.e., the turning angles and set speeds are physically fixed at zero, although the values in the program are something else. The idea is to increase and decrease the turning angles and set speeds until the angle flags and speed flags become true. This way the maximum values for the desired heading and desired speed can be found. The initialization process works as the state machine presented in figure 29.

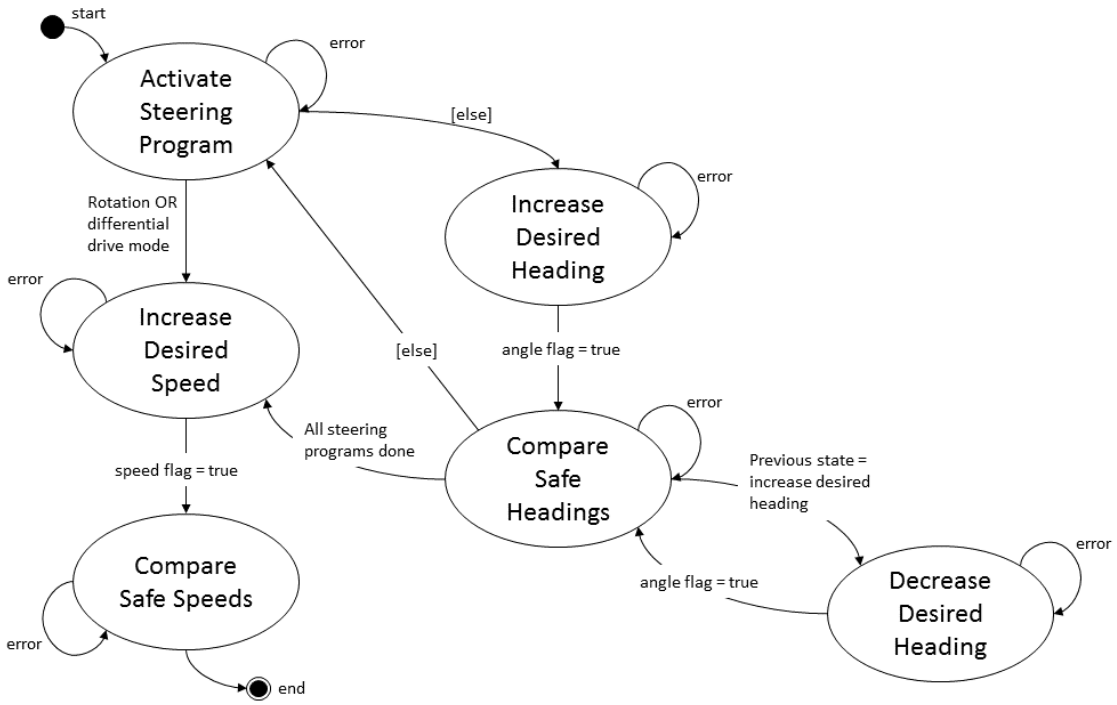

Figure 29. *UML state chart diagram of the initialization process.*

As mentioned before, in order for the rest of the reference value calculation FB to work, the initialization process must fabricate an active steering program, a desired heading and a desired speed. The desired speed and heading are initialized as zero, while the active steering program is set instantly at the first state, “activate steering program”. The steering program that is activated depends on what steering programs are enabled. If rotation mode or differential drive mode is enabled and is turned on using the switches on the remote controller, one of them are activated. It is important to remember that only one of either rotation mode or differential drive mode can be activated at once. If neither of rotation mode or differential drive mode is enabled, either counter-phase, front wheel, rear wheel or crabwise steering program is activated. When the rotation mode or differential drive mode is activated, the initialization process needs to run only for that particular steering program, because the operator has specifically chosen to drive in only that steering program at that time. If any other steering program is selected, the initialization process may have to run a second time after completion of the first activated steering program. Consider for instance the characteristics of the desired heading during crabwise steering and front wheel steering. The same heading may cause a steeper turning angle during front wheel steering than during crabwise steering. It is therefore enough if the initialization process is run only for the front wheel steering. If however rear wheel steering is enabled as well, the initialization process needs to be run for that steering program too, because rear wheel steering might have a smaller maximum allowed desired heading than front wheel steering, due to the possibility of asymmetric layout of the wheels. At the end of the “activate steering program”-state, the desired heading is set to the resolution of the desired heading and the desired speed is set to the resolution of the desired speed. If the

activated steering program was rotation mode or differential drive mode the next state is “increase desired speed” otherwise the next state is “increase desired heading”.

During the “increase desired heading”-state the desired heading is increased until an angle flag is raised. As soon as a flag is raised, the previous desired heading is saved as a safe value of the desired heading and the state is changed to “compare safe headings”, where the recently saved safe value is compared to previously saved safe values. If the recently saved safe value is smaller than the previously saved values, it will be used as the normalization constant $\max_{\alpha}$. After the comparison of the newly derived safe value and old safe values, the heading is reset and decreased gradually in the state “decrease desired heading”. Practically, the vehicle turns to the left instead of right as it did in the state “increase desired heading”. Again when an angle flag is raised, the state is changed to “compare safe headings” just as it was when an angle flag got raised during “increase desired heading”. When both the safe desired headings derived in “increase desired heading” and “decrease desired heading” have been compared in the state “compare safe headings”, the state is changed to “activate steering program” if there still are steering programs that need to be run through the initialization process and “increase speed” if all necessary steering programs have already been run through the initialization process.

During the “increase desired speed”-state, the desired speed is increased in the crabwise mode with all turning angles fixed at zero, until a speed flag is raised. When a flag is raised, the previous desired speed is saved as safe and the state is changed to “compare safe speeds”, where the saved safe speed is set as the normalization constant $\max_{\text{ref}}$. After the “compare safe speeds”-state, the initialization process has finished.

There are several different scenarios, in which the initialization process can fail. If for instance all steering programs are disabled, the “activate steering program”-state will fail. In that case, the state of the initialization process is stuck at “activate steering program” until the error is fixed, i.e., a steering program is enabled. When an error is corrected, the initialization process starts over. Upon normal completion of the initialization process, the calculated normalization constants scale the desired heading and speed such that the reference values will not cause any of the wheels to exceed their corresponding maximum speeds or turning angles.

4.2.6 Calculation of the Position of the ICR, Reference Velocity and Angular Velocity

The calculation of the location of the ICR, reference velocity and angular velocity is done using the equations derived in chapter 3. Which equations that are used, depend on which steering program is active at the time. The process is quite straight forward and can be best understood by carefully studying the equations in chapter 3.

4.2.7 Handling Raised Flags

If the initialization process has not ended as expected, there is a chance that a wheel may exceed either the maximum allowed turning angle or maximum allowed set speed. To prevent that from happening, the reference value calculation FB will stop changing the reference values such that it may increase the turning angle or set speed when the corresponding flag is raised.

As soon as a flag is raised the current desired heading or speed is saved as safe. If the angle flag is high, the current desired heading is saved as safe and if the speed flag is high the current desired speed is saved as safe. The principle for both the desired heading and speed is the same. As long as the angle flag is high, the desired heading will be fixed at the safe value despite the analog values coming from the joysticks try to increase the turning angles. When the analog joystick values finally would cause the turning angles to decrease, the desired heading is released from the fixed safe desired heading.

As long as the steering program does not change during a high flag, the situation is handled quite straight forward as explained just above. If the steering program however changes while a flag is high, things are a bit more complicated. The safe desired speed and heading can be something different in one steering program than in another steering program. The calculated desired speed and heading are therefore saved as set points at the moment when a steering program changes while a flag is high. The normal equations used in the normalization of the joystick values are skipped and the desired heading or speed is increased gradually by 10 % at every cycle until the set points are reached. This way the gradual increase will reset the flags and recreate the correct safe desired speed and heading for the steering program that has recently been activated.

4.2.8 Setting Outputs

If the execution of the reference value calculation FB has proceeded all the way to the step where the outputs are set, the preceding calculations has succeeded without errors. Before setting the outputs, it is important that the reference values are checked for invalid values such as infinity or NaN. The “REAL”-type variable in the Siemens STEP 7 (TIA Portal) programming environment is a single-precision floating point number following the IEEE754 standard. The invalid values in the standard are presented in table 4,

Table 4. *Invalid values in the single-precision IEEE754 standard.*

+∞	0 111 1111 1000 0000 0000 0000 0000 0000
-∞	1 111 1111 1000 0000 0000 0000 0000 0000
NaN	0 111 1111 1XXX XXXX XXXX XXXX XXXX XXXX
NaN	1 111 1111 1XXX XXXX XXXX XXXX XXXX XXXX

where X is either 0 or 1, but all X's cannot be 0. Looking at the values in table 4, it can be seen that all invalid values have one thing in common. The bits 1-8 all have the value "1", which means there is an opportunity to check for all invalid values at once. The process works as follows: The variable of type "REAL" is converted to "DWORD", after which it is ANDed with the value 0 111 1111 1000 0000 0000 0000 0000 0000. If the result of the AND operation is 0 111 1111 1000 0000 0000 0000 0000 0000, it means that the value of the "REAL"-type variable is invalid.

Although checking for invalid values is a very important step that needs to be done before finally setting the outputs, it is in the end supposed to function just as a safety precaution. It should never come to that, that a reference value is invalid. The reference value calculation FB is designed such that invalid values should not be possible in normal operation. The size of the vehicle is for instance checked already at the input checks step and all divisions by zero are checked and handled at the places where such may occur. If there for some reason however is an invalid value in the reference values, the vehicle is set in idle mode to prevent damaging the drive and steer motors.

4.3 Structure of the Set Value Calculation FB

The set value calculation FB has intentionally been made much shorter than the reference value calculation FB, in order to achieve a more optimal program structure. Because the main idea of the program structure is to use only one reference value calculation FB and an unrestricted amount of set value calculation FBs, it is reasonable to do the heavy calculations only once in the reference value calculation FB and do only the very essential calculations in the set value calculation FBs. This way the program structure produces the least possible load on the processing unit in the PLC.

The main task of the set value calculation FB is to calculate the set speed and turning angle of a single wheel positioned anywhere in a Cartesian coordinate system. The structure of the FB is in general quite similar as the reference value calculation FB as can be seen in figure 30.

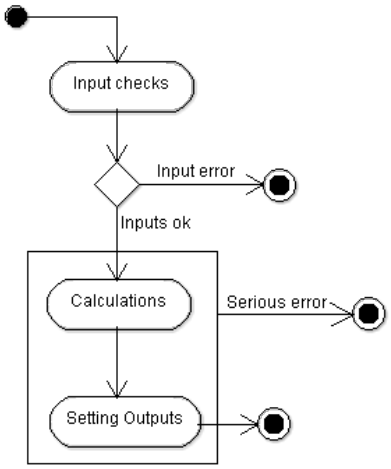

Figure 30. *UML activity diagram of the set value calculation FB.*

Just as in the reference value calculation FB, the set value calculation FB starts with checking the inputs for errors. If there is an error in the inputs such as the x-coordinate of the wheel is unrealistically large etc. the execution of the FB is terminated immediately. In addition to errors detected during the input checks, the set value calculation FB is terminated immediately if there is a serious error during the execution of the rest of the FB just as in the reference value calculation FB. The "Setting Outputs" step is also quite similar in the set value calculation FB as in reference value calculation FB. Therefore, the input check and setting outputs steps won't be handled here a second time. The main part of the set value calculation FB is the "Calculations" step, which is explained in the next section.

4.3.1 Calculations

The calculations step is divided into five different sections according to the active steering program. When idle steering program is active, the set speed and turning angle is fixed at zero. In the rest of the steering programs the calculations are done according to the equations derived in chapter 3. In addition to the equations presented in chapter 3 some intelligence has been added to the FB. If the rotation mode steering program is active and the turning angle exceeds either the maximum positive or negative limit of the turning angle, 180° is added or subtracted from the turning angle and the speed is negated to test if that turning angle would be possible instead for instance. If that angle however also exceeds one of the limits, the rotation mode is clearly not possible.

After using the equations presented in chapter 3, the range of the turning angle in some of the steering programs is [-90°, 90°] although the turning angle should be outside that range. The reason is that the $\tan^{-1}$ function is defined in that range. The calculated turning angle can however be corrected by comparing the sign of the heading to the sign of the calculated turning angle. Therefore if for instance the calculated turning angle is -50°, while the direction of the heading is positive, 180° must be added to the calculated value

after which the correct turning angle, i.e., 130° is gotten. After the correct turning angle has finally been derived, the turning angle and set speed is checked, so that they don't exceed the limiting values. If they do, the corresponding flag is raised. If they don't, the flags remain low.

The set speed and turning angle that are outputted from the set value calculation FBs are forwarded to be handled in other parts of the software in the vehicle not made in this thesis.

4.4 Proof of Concept with Simulink Simulations

Because the vehicles that the program structure will be implemented in will be handling loads of several hundred tons, it is important that the program structure is well tested before used in a real vehicle. During the first steps of this thesis, the physical equipment was not available, thus it was decided to do the first tests in a virtual environment. After completion of this thesis it may be stated that the simulations done in the virtual environment were highly useful and that errors found during that stage may not have been found as easily with the physical equipment.

The first simulations that were done, where simple MATLAB animations, but soon it was noticed that animations were not quite enough for testing all the desired features of the program structure. It was found out that Xbox 360 controllers can easily be connected to a PC (Personal Computer) and Simulink. Following that path the simulation environment was created in Simulink. The created Simulink model is presented in figure 31. The purpose of the figure is not to display all variable names and minor details, but to show how the presented block structure was implemented in the virtual environment.

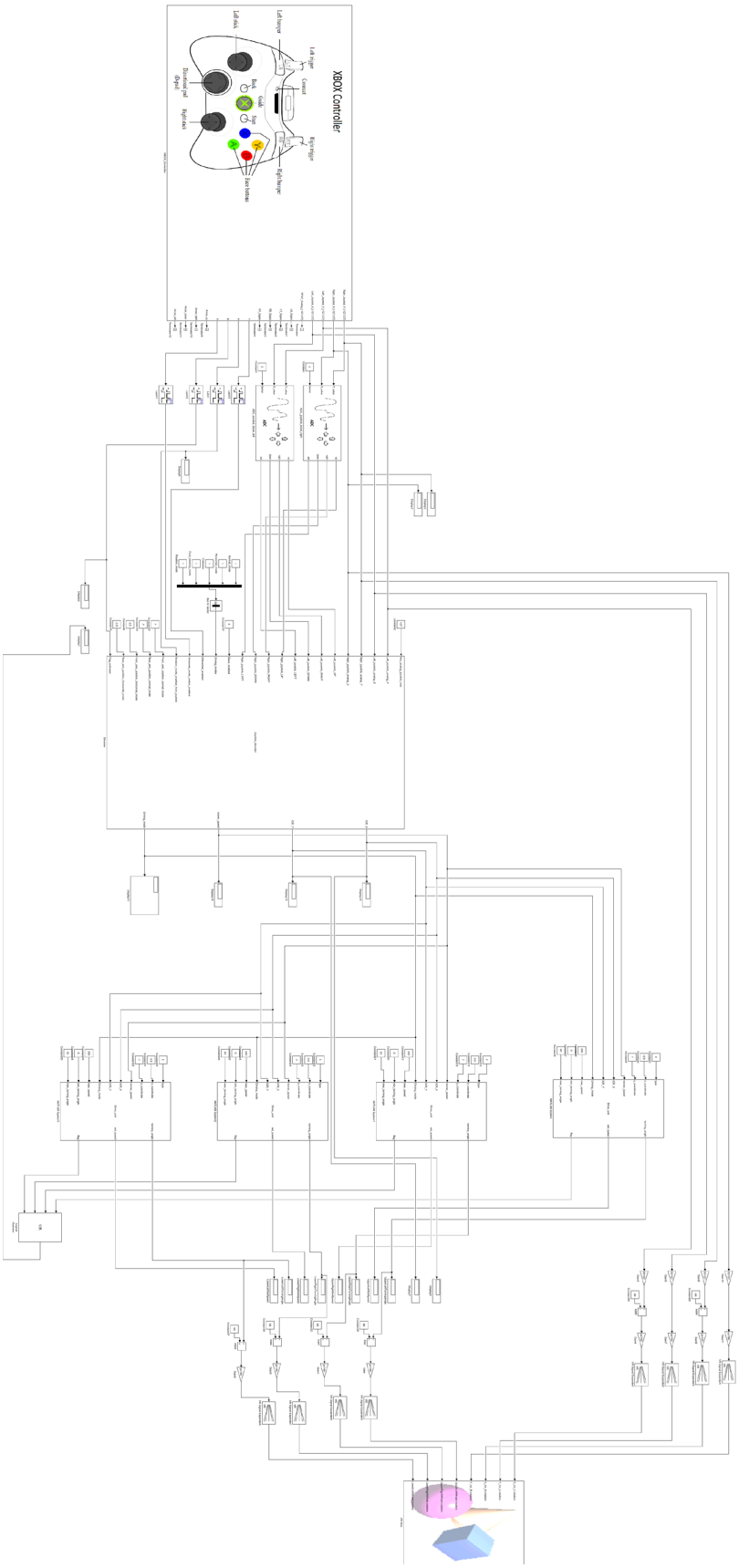

Figure 31. *Simulink model of the simulated program structure.*

On the left side of the Simulink model is a block that reads the values coming from the Xbox controller. The reference value calculation FB is located approximately in the middle getting its inputs from the Xbox controller. The reference values are divided to four set value calculation FBs, which means that the simulated vehicle is a four-wheel vehicle. The set values calculated by the four set value calculation FBs are further sent to a virtual world sink. The analog joystick values are also sent to the virtual reality sink to express the joystick positions. The virtual world connected to the virtual world sink in the Simulink model was created using the V-Realm Builder v2.0, which is a MATLAB extension. The virtual world created in the V-Realm Builder is presented in figure 32.

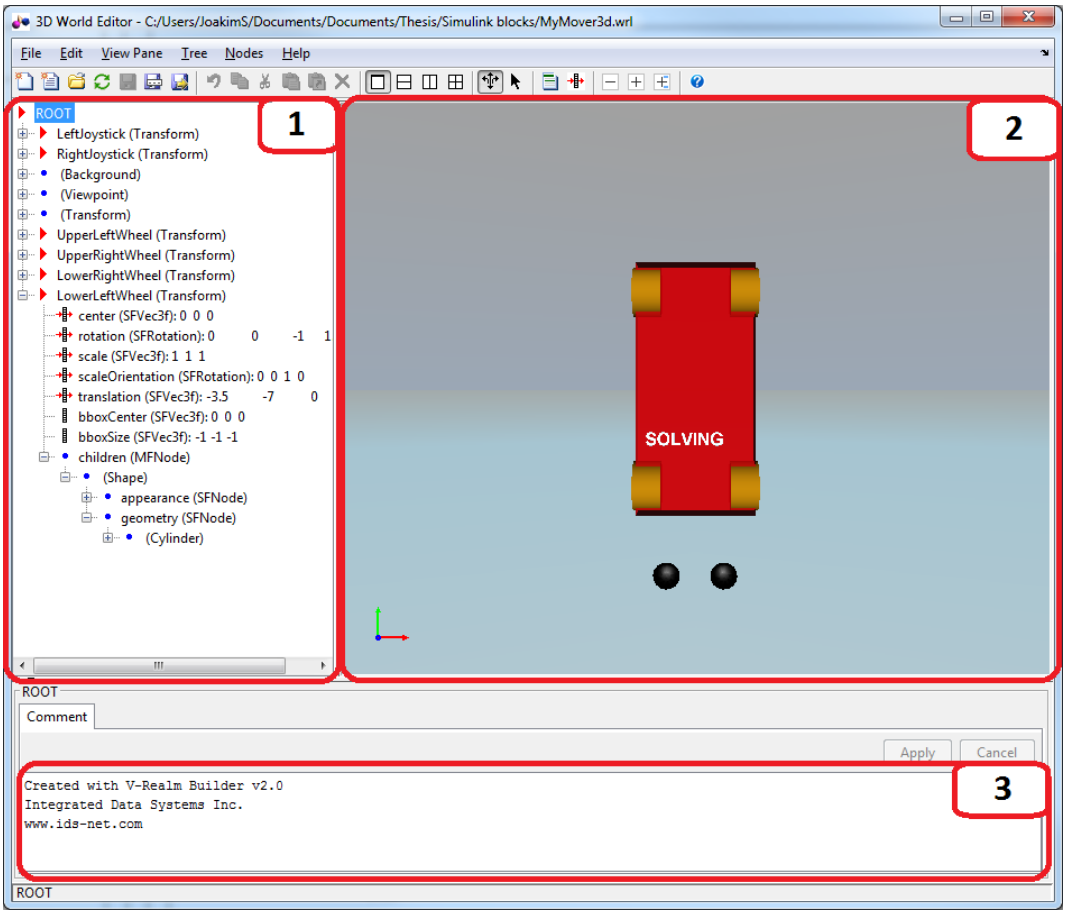

Figure 32. *Virtual world created in V-Realm Builder. The Virtual Reality Modeling Language (VRML) can be seen in section 1, the virtual world in section 2 and the transcript pane in section 3.*

The virtual world consists of the four-wheel vehicle and two joysticks. The vehicle frame is the red rectangle in section 2 in figure 32, while the wheels are drawn as yellow cylinders at each corner of the vehicle frame. The virtual world is created in the V-Realm Builder using the Virtual Reality Modelling Language (VRML) that can be seen in the left pane, section 1 of the window in figure 32. The joysticks and the wheels are transform-type objects, which means that the position, orientation and other attributes related to the object can be modified. The basic attributes of the object can be set to a default by adding children to the transform object. Each wheel has for instance a shape-type child

with the geometry of a cylinder, which can also be seen by carefully studying the left pane in section 1 of figure 32. The attribute of the wheels that is modified in the Simulink model is the rotation about the z-axis. The purpose of this section is not to pursue the details of the VRML, the important thing to understand here is that the signals in the Simulink model connected to the virtual reality sink can be used to control a few attributes of the transform objects in the virtual world. The Simulink model and the virtual world during a running simulation is presented in figure 33.

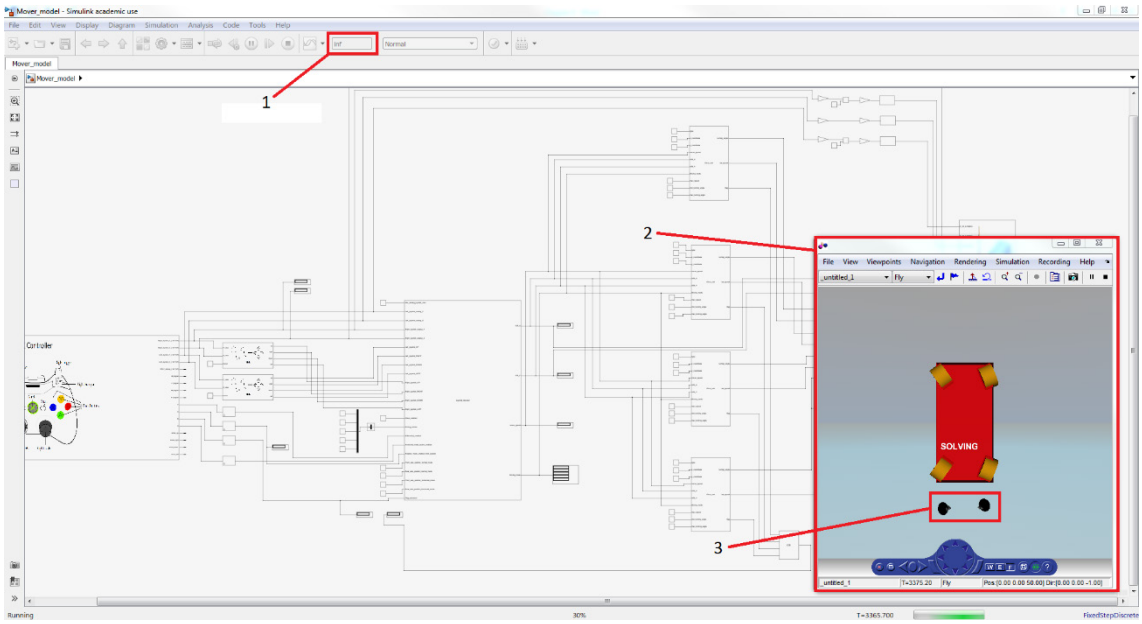

Figure 33. *Running simulation. 1: Simulation time, 2: Virtual world, 3: Virtual model of the joysticks on the Xbox controller.*

As can be seen in figure 33, the simulation time of these types of simulations is usually set to infinity, which means that the simulation will run until it's stopped. In the particular captured situation presented in figure 33, the four-wheel vehicle is turning in the counter-phase manner. The left joystick is pointing to the left, i.e., the cardinal direction W and the cardinal direction of the right joystick is NE. These types of tests were done to visually confirm that the position of the joysticks activated the correct steering program and that the relationship between the turning angles between each wheel seemed to follow the mathematical equations derived in chapter 3.

The Simulink model, although a very important part of the development of this thesis, was in the end just a primitive version of the final program structure. Several features could unfortunately not be tested properly in the virtual model. Therefore the details of the Simulink model is left a bit vague, to avoid emphasizing the importance of the simulations. The Simulink model served merely as a proof of concept of the program structure, while the most important tests, confirming the requirements were done in the final programming environment, namely Siemens STEP 7 (TIA Portal).

5. TEST RESULTS

The main goal of this thesis was always to build a program structure that would be tested and ready to use in Solving's products after the completion of the thesis. Tests with the real physical equipment were therefore done in addition to the tests in the virtual environment. To ensure that the programming style and structure overall fulfills the requirements given by Solving, the Siemens programming guideline and styleguide (Siemens 2017) were followed during the entire programming phase of this thesis. The guideline and the styleguide has recently been adopted by Solving to create better consistency between Siemens programs. When all software engineers follow the same style and program structure, it's easier to understand one another's code and the program is much more reliable, mainly because there are several ways of doing things in the Siemens environment and the best approaches have been presented by Siemens themselves in the guideline and styleguide. From the viewpoint of the management at Solving it is beneficial if all software engineers use the same style and program structure, because it is then easier to change the responsible software engineer on a project.

The testing with the physical equipment of this thesis can be divided into three parts. In the first part, the equations derived in chapter 3 were tested as such in an electric modular transporter that was made for the train industry. In the second part, the program structure presented in chapter 4 was implemented in Siemens STEP 7 (TIA Portal) and tested with the simulation tools. Finally in the third part, the new program structure was implemented in the software of the Solving air film movers. All the tests were successful and in some cases the response of the program structure was even better than expected.

5.1 Testing the Concept in an Electric Modular Transporter Made for the Train Industry

In the beginning of the programming phase of this thesis, Solving was conveniently developing a new type of electric modular transporter for the train industry. The vehicle was intended mostly for demonstrational purposes, which made it a perfect test subject for the new stuff developed in this thesis. The vehicle is equipped with two drive units, one front drive unit and a rear drive unit. Both drive units are equipped with a steer and a drive motor. The drive units were positioned on the left side of the vehicle center and displaced a bit in relation to one another. The bottom view of the vehicle is presented in figure 34.

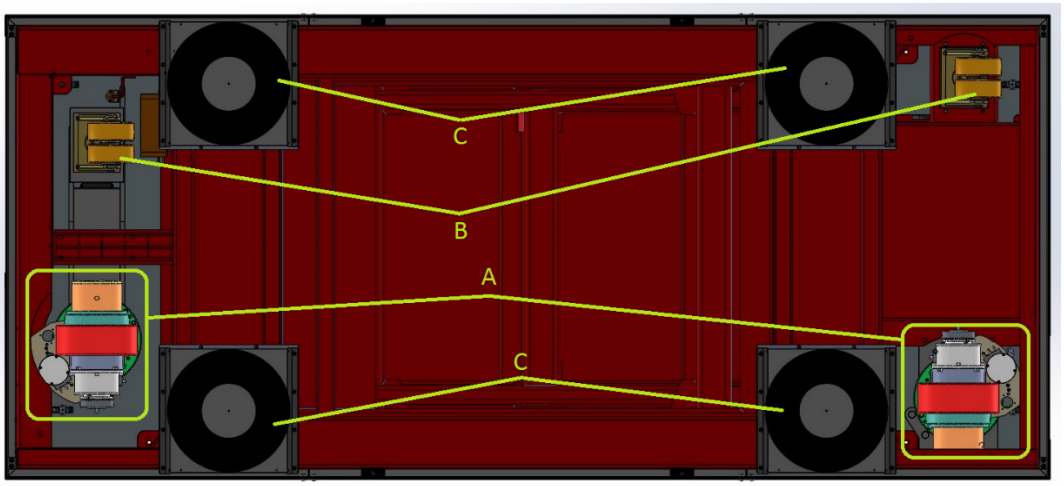

Figure 34. *Bottom view of Solving electric modular transporter for the train industry. A: Front and rear drive unit, B: Swivel wheels, C: Air bearings.*

The bottom view of the vehicle in figure 34 was particularly chosen in order to show all the wheels on the vehicle. This way it can be seen that the vehicle has four air bearings, two swivel wheels and two drive units. As mentioned earlier Solving did not before this thesis have a structure for calculating the speeds and turning angles for the wheels on vehicles such as the one presented in figure 34. The problem is the horizontal displacement between the wheels. The equations presented in chapter 3 should and did however overcome this challenge.

Once again, just to be clear, the program structure presented in chapter 4 was not implemented in the vehicle presented in figure 34. Because the vehicle is merely intended for demonstrational purposes and the load handling capacity of the vehicle is quite low, the requirements on the program were much lower on this vehicle than on the final program structure. Only three different steering programs, i.e., counter-phase steering, crabwise steering and differential drive mode were implemented on the vehicle. During the counter-phase and crabwise steering programs, the range of the turning angle of each wheel was set to about [-30°, 30°], while the turning angle during differential drive mode is fixed at +90°. Because the range of the turning angle is quite narrow there was no need to handle exceptions that happen when the turning angle goes outside the range [-90°, 90°], which is another factor that simplified the structure of the program.

The steering configuration, i.e., how the steering programs are changed was also made different in the vehicle presented in this section than in the final program structure. Because only three different steering programs were implemented, it was logical to make a custom steering configuration. The counter-phase steering program is active by default, meaning that as long as the right joystick does not have a horizontal direction, the counter-phase steering program is active. If the horizontal direction of the left and right joystick is the same, the crabwise steering program is activated. The differential drive mode on the other hand is activated by a switch on the remote controller. By turning the switch,

the wheels turn from 0° to +90°, after which the speed of the rear wheel is controlled with the right joystick and the speed of the front wheel is controlled with the left joystick.

The tests proved that the turning angle and speed of the front and rear wheel behaved exactly as expected. The response of the joystick movements were even perhaps better than expected at first. Meaning that the delay between the joystick movement and the actual response, i.e., acceleration or turning of the vehicle was much smaller than initially expected.

5.2 Presentation of Function Blocks and Simulations done in Siemens STEP 7 (TIA Portal)

After the program structure presented in chapter 4 had been tested in a neutral environment, i.e., Simulink and it had been verified that it indeed is possible to do calculations as the ones presented in chapter 3 in the Siemens STEP 7 (TIA Portal) environment, it was quite clear that the next step would be to implement and test the actual program structure in the Siemens environment. During the testing of the vehicle presented in the previous section it was noticed that there are several factors not related to the results of this thesis that have a disturbing effect on the behavior of the vehicle. The controllers following the set value calculation for instance, cause a certain inevitable slowness. Because of such disturbances, the program structure was at first tested only with the Siemens S7-PLCSIM (TIA Portal) simulation tool. This way the test results were not subject to disturbances caused by factors not related to the new program structure.

5.2.1 Reference Value Calculation FB

The final program structure consisting of the two different types of FBs will be presented in the following paragraphs. The reference value calculation FB is presented first of the two types. The reference value calculation FB presented in figure 35 should look familiar as the general block structure was already presented in chapter 4.

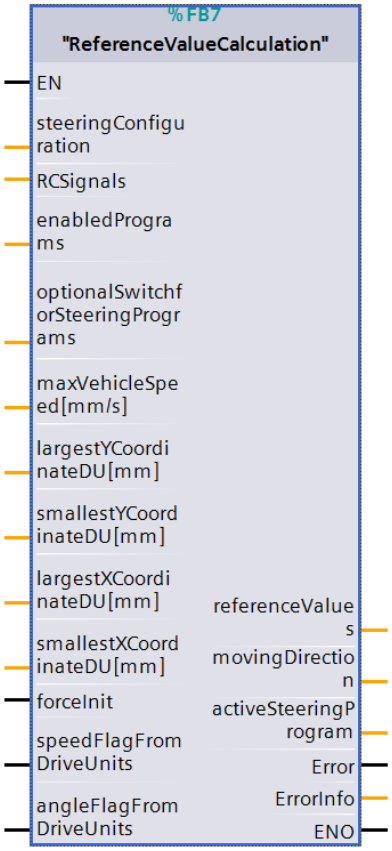

Figure 35. *Reference value calculation FB.*

In order to compress the size and clarify the interface of the reference value calculation FB, PLC data types has been used for the inputs and outputs of the FB. The PLC data type resemble the old “Struct”- data type, where several different variables can be saved under the same data type. Structs are still available in Siemens PLC programming languages, but nowadays it is recommended to use PLC data types instead, because they are more addressable for changes. The benefits of using PLC data types should be quite obvious. There is no need for the software engineer using the FB to know what all different parameters included in the “referenceValues”-variable are for instance. All inputs and outputs of the “ReferenceValueCalculation” FB are listed below:

steeringConfiguration

Type: Int

Determines the way at which a steering program is changed. 0 is the default steering configuration used in Solving air film movers today. 1 is a modified version of the default configuration, where the turning angle is not negated when the moving direction of the vehicle changes unlike the behaviour in the default configuration. The steering configuration 2 uses the Boolean values defined in the input “optionalSwitchforSteeringPorgrams” to activate/deactivate steering programs. All integers higher than 2 have been reserved for custom steering configurations that can be made by skilled software engineers, due to the “CASE”-structure done in the internal SCL code of the FB.

RCSignals
Type: “RemoteControllerSignals” (PLC data type)
A PLC data type containing the value of the maximum analog joystick reading, the analog and digital joystick values and Boolean values corresponding to the position of the switches for crosswise mode, rotation mode and differential drive mode on the remote controller.

enabledPrograms
Type: “EnabledPrograms” (PLC data type)
A PLC data type containing all the steering programs and the crosswise mode. A steering program or the crosswise mode is enabled when the corresponding value is true.

optionalSwitchforSteeringPrograms
Type: “SteeringPrograms” (PLC data type)
Input used in steering configuration 2 for activating/deactivating steering programs.

maxVehicleSpeed[mm/s]
Type: Real
The maximum desired vehicle speed. The maximum speed is determined in a situation where the vehicle is moving straight and all the points fixed to the vehicle body have the same speed.

largestYCoordinate[mm]
Type: Real
The largest y-coordinate of all the wheel planes. This is in other words the distance between the front axis and the vehicle center when the vehicle is driving in the lengthwise direction.

smallestYCoordinate[mm]
Type: Real
The smallest y-coordinate of all the wheel planes. This is in other words the distance between the rear axis and the vehicle center when the vehicle is driving in the lengthwise direction.

largestXCoordinate[mm]
Type: Real
The largest x-coordinate of all the wheel planes. This is in other words the distance between the front axis and the vehicle center when the vehicle is driving in the crosswise direction.

smallestXCoordinate[mm]
Type: Real
The smallest x-coordinate of all the wheel planes. This is in other words the distance between the rear axis and the vehicle center when the vehicle is driving in the crosswise direction.

forceInit
Type: Bool
Input for the “forceInitialization” signal coming from the set value calculation FBs. Please note that if there are several set value calculation FBs, the “forceInitialization” signals from all the set value calculation FBs must be logically ORed as shown in chapter 4.

speedFlagFromDriveUnits
Type: Bool
Input for the “speedFlag” signal coming from the set value calculation FBs. Please note that if there are several set value calculation FBs the “speedFlag” signals from all the set value calculation FBs must be logically ORed as shown in chapter 4.

angleFlagFromDriveUnits
Type: Bool
Input for the “angleFlag” signal coming from the set value calculation FBs. Please note that if there are several set value calculation FBs the “angleFlag” signals from all the set value calculation FBs must be logically ORed as shown in chapter 4.

referenceValues
Type: “ReferenceValues” (PLC data type)
The reference values needed in the set value calculation FBs. The parameters of the “ReferenceValues” PLC data type are listed in table 5.

Table 5. *Parameters in the "ReferenceValues" PLC data type.*

Signal	Type	Comment
initialization	Bool	true = initialization process is running
activeSteeringProgram	"SteeringPrograms"	Indicates what steering program is active
idle	Bool	steering program
crabwise	Bool	steering program
frontWheelSteering	Bool	steering program
rearWheelSteering	Bool	steering program
counterphaseSteering	Bool	steering program
rotationMode	Bool	steering program
differentialDrive	Bool	steering program
crosswiseEN	Bool	true = crosswise mode is activated
headingFront	Real	Heading of the vehicle front
headingMid	Real	Heading of the vehicle mid
headingRear	Real	Heading of the vehicle rear
xCoordinateOfICR	Real	x-coordinate of the ICR
yCoordinateOfICR	Real	y-coordinate of the ICR
refAngularVelocity	Real	The angular velocity of the vehicle
refSpeed	Real	The speed of the vehicle center

The "referenceValues" output must be connected to the corresponding inputs of the set value calculation FBs in order for the block structure to work.

movingDirection
Type: String
For debugging purposes only. The value of this parameter will be either "FORWARD", "BACKWARD" or "NONE".

activeSteeringProgram
Type: String
For debugging purposes only. The active steering program in String-form.

Error
Type: Bool
Expressing if there was an error during the execution of the FB. (true = error, false = ok).

ErrorInfo
Type: Int
0 if everything is ok. If there is an error, this output will have the value of one of the error codes presented in appendix A.

The PLC data types reduce the size of the FB significantly. The “RemoteControllerSignals”-PLC data type does for instance contain all the essential signals such as the analog and digital joystick signals and the Boolean values expressing the position of a few of the switches on the remote controller. The name of all the inputs and outputs follow according to the aforementioned styleguide camel case notation. Meaning that the first letter in the variable name is written in small casing and if the variable consist of several words the first letter of all the following words are written in capital casing.

If there is an error in the inputs or an error occur during the calculations done in the reference value calculation FB, the FB needs to inform the user about the error. The error procedure is quite simple. If there is an error, the “Error”-output is set true and an error code is displayed in the “ErrorInfo”-output. The error codes for the reference value calculation FB are presented in appendix A. The main idea of the error codes is that the higher the value of the error code, the more serious the error is. The error 1102 is for instance used just to inform the user that it is not possible to activate front wheel steering because it is disabled, while the error code 5230, with a much higher value than 1102, is a sign of invalid reference speed.

5.2.2 Set Value Calculation FB

Although the function of the reference value calculation FB and set value calculation FB are completely different, some basic features have been tried to keep consistent between the two FBs. A Boolean error output and an output for the error code has been used in the set value calculation FB just as in the reference value calculation FB. The “ReferenceValues”- PLC data type that was presented as one of the outputs of the reference value calculation FB has of course also been used for the reference value input of the set value calculation FB. The set value calculation FB is presented in figure 36.

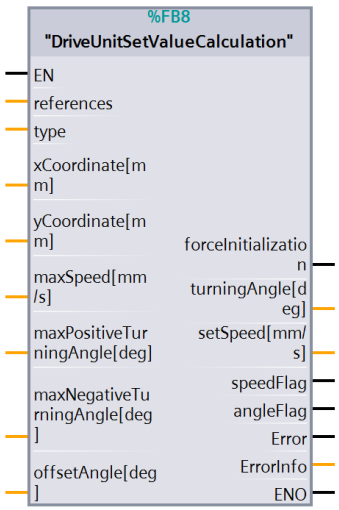

Figure 36. *Set value calculation FB.*

Remember that the recommendation is to connect the ENO output of the reference value calculation FB to the EN input of the set value calculation FBs. This way the reference value calculation FB will enable the set value calculation FBs and if there is a serious error in the reference value calculation FB, the ENO output will turn false, which on the other hand leads to that all set value calculation FBs will get disabled. Just as the inputs and outputs of the reference value calculation FB were presented earlier, the inputs and outputs of the "DriveUnitSetValueCalculation" FB are listed below:

references
Type: "ReferenceValues" (PLC data type)
Input for the reference values coming from the "ReferenceValueCalculation" FB. See section 5.2.1 for a more detailed description.

type
Type: Int
Type of the drive unit corresponding to the wheel plane. 0 = S/D (Steer/Drive), 1 = S, 2 = D. If the type is S, the "setSpeed[mm/s]" will be fixed at 0 and if the type is D, the "turningAngle[deg]" will be fixed at 0.

xCoordinate[mm]
Type: Real
The x-coordinate of the wheel plane.

yCoordinate[mm]
Type: Real
The y-coordinate of the wheel plane.

maxSpeed[mm/s]
Type: Real
Maximum allowed speed of the wheel plane. After a successful run of the initialization process, the maximum allowed speed of a wheel plane should never be exceeded.

maxPositiveTurningAngle[deg]
Type: Real
Maximum allowed turning angle in the positive direction, i.e., to the right, clockwise direction.

maxNegativeTurningAngle[deg]
Type: Real
Maximum allowed turning angle in the negative direction, i.e., to the left or counter clockwise direction. Please note that the sign of this input is often negative.

offsetAngle[deg]
Type: Real
This value will be added to the “turningAngle[deg]” output. Can be useful if it is desired to drive the vehicle in a diagonal manner.

forceInitialization
Type: Bool
If this output becomes true, a new run of the initialization process must be done. This output is to be connected to the “forceInit” input of the “ReferenceValueCalculation” FB.

turningAngle[deg]
Type: Real
The calculated set value of the turning angle to the wheel plane with the coordinates [xCoordinate[mm], yCoordinate[mm]].

setSpeed[mm/s]
Type: Real
The calculated set speed to the wheel plane with the coordinates [xCoordinate[mm], yCoordinate[mm]].

speedFlag
Type: Bool
If this output becomes true it means that the “setSpeed[mm/s]” output is exceeding the maximum allowed speed in either the positive or negative direction. This output is to be connected to the “speedFlagFromDriveUnits” input of the “ReferenceValueCalculation” FB.

angleFlag
Type: Bool
If this output becomes true it means that the “turningAngle[deg]” output is exceeding the maximum allowed turning angle in either the positive or negative direction. This output is to be connected to the “angleFlagFromDriveUnits” input of the “ReferenceValueCalculation” FB.

Error
Type: Bool
Expressing if there was an error during the execution of the FB. (true = error, false = ok).

ErrorInfo
Type: Int
0 if everything is ok. If there is an error, this output will have the value of one of the error codes presented in appendix B.

One interesting choice that has been made, is the unit of all angles. It was decided to use degrees for all visible angles as those are much more intuitive than radians to comprehend, while the FB must internally use radians in order for the trigonometric functions in SCL to work.

The most essential result of the program structure as a whole is of course the “turningAngle[deg]” and “setSpeed[mm/s]” outputs of the set value calculation FBs. Those outputs are used later in subsequent parts of the vehicle software that are not done in this thesis. Remember also that the purpose of the three flags, i.e., “forceInitialization”, “speedFlag” and “angleFlag” is to inform the reference value calculation FB when to lock the value of the reference values, which is why these outputs must be fed back to the reference value calculation FB. Although one of the outputs of the set value calculation FB is “ErrorInfo”, just as one of the outputs of the reference value calculation FB, these outputs differ from one another. The main idea is the same, i.e., the higher the value of the error code, the more serious error, but the purpose is different. The meaning of the set value calculation FB error codes are presented in appendix B.

5.2.3 Simulations in Siemens PLCSIM

A variable amount of set value calculation FBs and a reference value calculation FB were used to create different types of vehicles to simulate and test the program structure as much as possible before implementing it in a real vehicle. Particularly four, five, seven, ten and twelve wheeled vehicles, but also other types of vehicles were simulated. With symmetrical wheel layouts, it was easy to verify the operational requirements simply because there are wheels that will have the same set speeds and turning angles when the layout is symmetrical. Asymmetric layouts were of course also tested. A typical simulation setup is presented in figure 37.

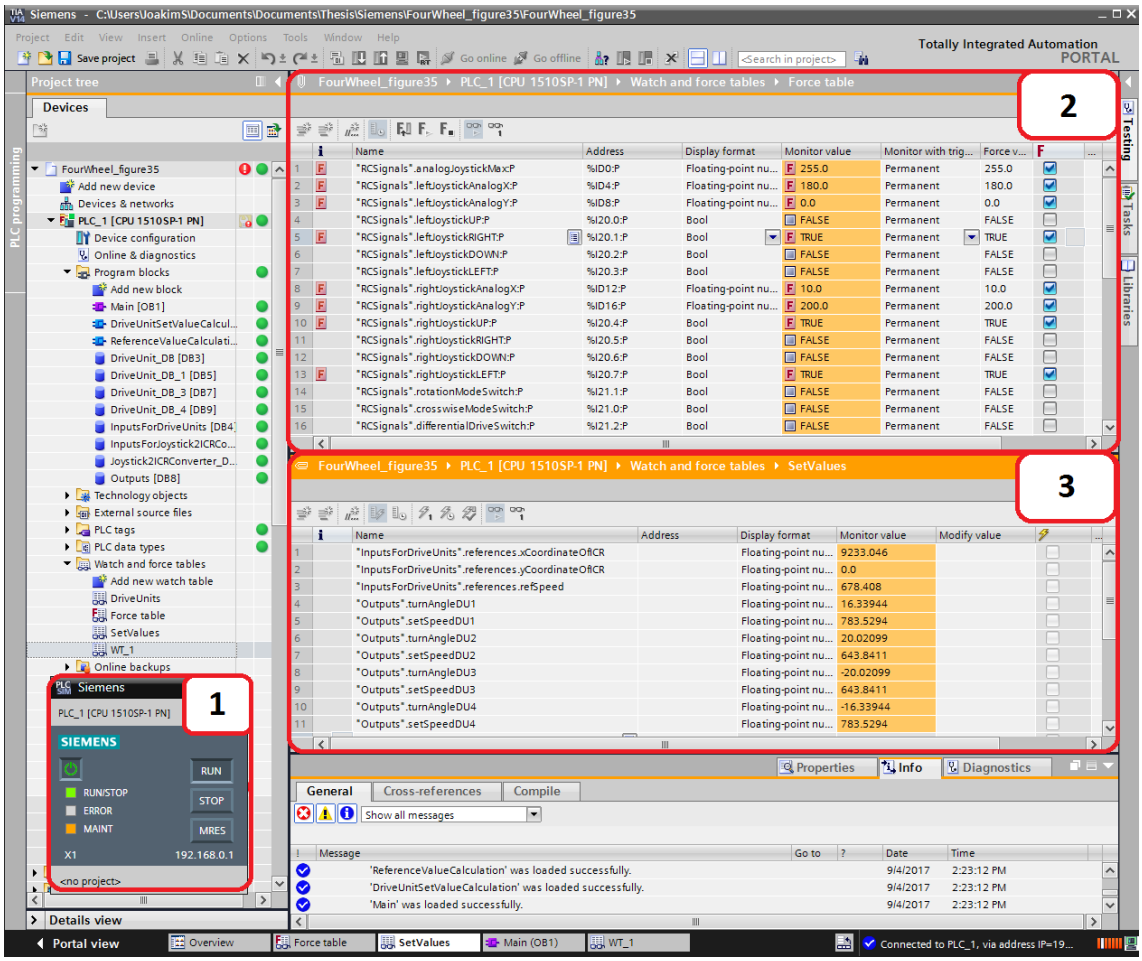

Figure 37. *Simulation of a four-wheel vehicle. 1: PLCSIM in run mode, 2: Force table forcing the right joystick to point to NW and the left joystick to E, 3: Watch table for monitoring the calculated turning angle and set speed of all wheels on the vehicle.*

In the bottom left corner of figure 37 it can be seen that the simulation is running and that the simulated PLC is in “RUN”-mode indicated by the green light. All the program blocks including the “ReferenceValueCalculation” and “DriveUnitSetValueCalculation” FBs are listed under the “Program blocks”-folder in the left pane in the window of the figure 37. The vehicle that is simulated in figure 37 is a four-wheel vehicle meaning that one of the “ReferenceValueCalculation” FBs and four of the “DriveUnitSetValueCalculation” FBs have been used in Main (OB1) as instructed before. In addition to the FBs and Main (OB1) there are a few Data Blocks (DBs) listed under the “Program blocks” as well, but they have only been added for structural purposes, which is why the purpose of the DBs will not be covered here.

To simulate actual change in physical inputs in Siemens TIA Portal, the force table like the one presented in figure 37 is used. In this case the force table was used to simulate the signals coming from the remote controller. In the particular captured situation in figure 37, the horizontal analog value of the left joystick is 180/255 and the vertical value is 0/255. The only digital value of the left joystick that has been forced true is the RIGHT

value, which means that the left joystick is pointing straight to the right. The right joystick on the other hand is pointing to the upper left corner in the joystick space, because the digital directions UP and LEFT are true and the RIGHT and DOWN values are false. The vertical analog value of the right joystick is 200/255 and the horizontal analog value of the right joystick is 10/255.

The corresponding results of the fabricated joystick values can be seen in the watch table in the lower pane on the right side of figure 37. Because the left joystick is pointing to the right and the right joystick is pointing to the upper left corner of the joystick space, the counter-phase steering program is active, which can be seen on the first row of the watch table and by studying the values of the turning angles and set speeds. The turning angle of the upper left wheel, denoted as DU1 in figure 37 is for instance 16.33944° and the turning angle of the bottom left wheel, DU4 is -16.33944°, which is exactly what the counter-phase steering program is supposed to do. The name of the counter-phase steering program can be further emphasized here by the different signs, i.e., phases of the turning angles on wheels on the same side.

By similar situations as the one presented in figure 37, different joystick values were simulated and tested with the program structure in order to compare the set values computed by the program structure with desired calculated values. Because pure numerical values that are not disturbed by outside factors can be seen in the simulation environment, it was a more powerful debugging tool than tests with the actual equipment. Tests with the actual equipment was more of a visual confirmation than comparisons between computed values and desired calculated values. Simulations like the ones presented in this section were however very time consuming, mainly because all values that were to be altered were always to be altered manually.

5.3 Implementing the Program Structure in Solving Air Film Movers

Although the operation of the program structure had been numerically verified as explained in the previous section, it was very important to test it in a real vehicle as well. Things that cannot be noticed in a simulation environment are for instance undesired slipping or jumping, acceleration, response time and even turning and moving directions. The sensors or motors on the drive units could for instance be installed wrong which results in the motor running in the wrong direction.

During the first tests with the real vehicles, a particular problem that was found caused the drive units to increase speed during turns in order to keep the speed of the vehicle center constant, although the analog joystick reading of the right joystick remained unchanged. Changing steering programs could additionally cause the speed of the wheels to behave in an undesired way. This is just an example of what types of problems that was

found during the tests conducted with the real vehicles. It is clear that these types of problems, related to the behavior of the vehicle were difficult to find in the simulation environment, but easy to detect visually with a real vehicle. In the final version of the program structure all of these types of problems have of course been fixed.

The main idea of the vehicle software in Solving air film movers is to divide the Main (OB1) program into modules, which all handle a specific set of operations on the vehicle. This way only the most important information is visible in Main (OB1), while the detailed internal implementation of each operation is seen only when a network in Main (OB1) is opened. This is a very common structure in software design nowadays and is also recommended in the aforementioned Siemens guidelines (Siemens 2017). The purpose of dividing the Main (OB1) program into modules is to make things easier for people reading the code. If there is for instance a problem with the warning light at the front of the vehicle, the software engineer reading the code, can simply go to the network handling lights in Main (OB1). If necessary, the network can be opened to further study the internal implementation of how and when the lights are turned on or off. This way there is no need to study other networks in Main (OB1) and the software engineer saves time by focusing only on the part of the code that is essential. Also before this thesis, there was a network in Main (OB1) handling the set value calculation of turning angle and speed for each wheel. A new network consisting of the solution in this thesis was built to replace this old network. The new network in Main (OB1) consist of a simple FB that acts as an interface for the program structure presented in this thesis. Using the interface it is possible to do all kinds of things such as change the coordinates of the drive units on the mover and limit the maximum vehicle speed.

Current Solving air film movers are built with only two drive units. If the program structure would have been tested with a vehicle with only two wheels that are moreover aligned, several features of the new program structure would have been left untested. Fortunately for the results of this thesis there is a possibility to connect air film movers in tandem. One of the movers are then called a master and the other is called slave. The computations and control are done in the master mover in such a way that the remote controller of the master is used to operate both the master and the slave moving in unison. In the favor of this thesis, the vehicles could be placed at different horizontal and vertical distances to one another in order to test the behavior of different types of wheel layouts used in different types of vehicles. Most common master and slave configurations wanted by customers are the side-by-side and end-to-end options, but in the light of this thesis also other configurations were tested. For instance T-formations, placing the vehicles diagonally to one another and other random configurations. A typical test scenario is presented in figure 38.

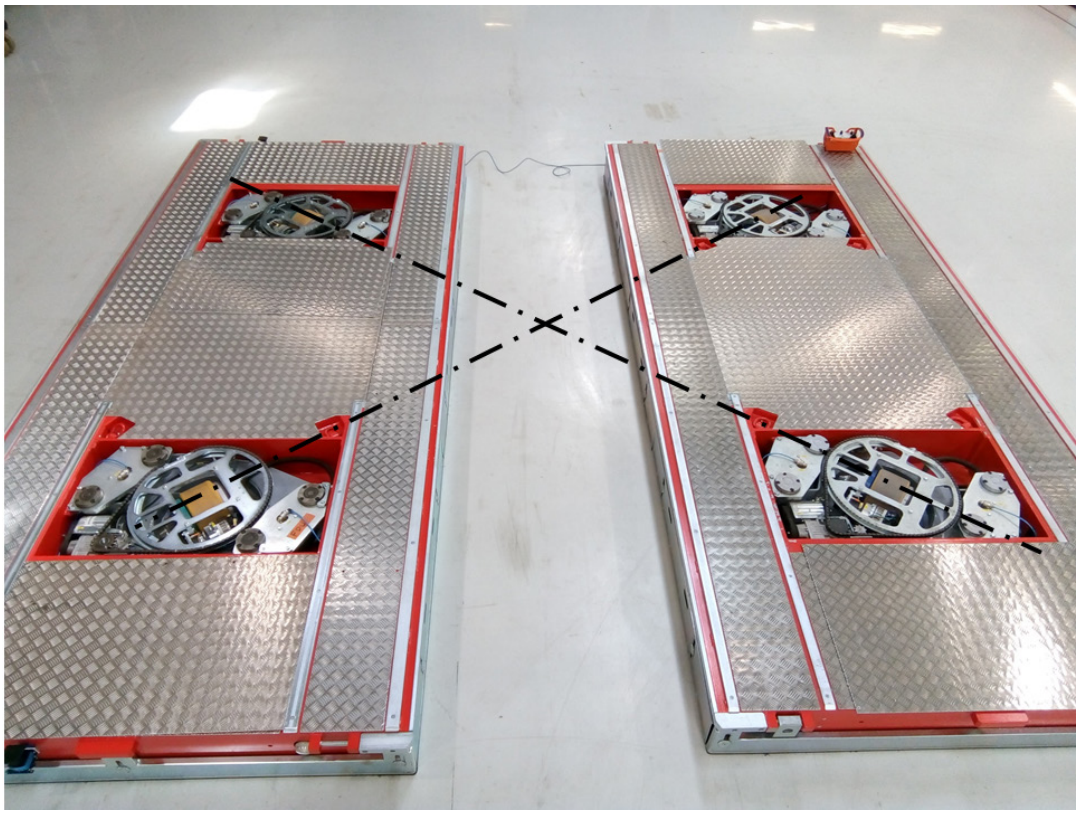

Figure 38. *Master (right) and slave (left) air film movers driving in rotation mode. Normal lines to all wheel planes marked with dot-dashed line, intersecting at the vehicle center.*

The master-slave combination in figure 38 is clearly driving in the rotation mode steering program and just to highlight that the normal lines to all the wheel planes in fact intersect at the ICR, which in this case is the vehicle center, the normal lines to all wheel planes have been drawn as dot-dashed lines to the figure. A few other interesting things in figure 38 is the radio remote controller in the upper right corner, placed on the master and the Ethernet-cable connecting the master and slave, responsible for providing the communication between the vehicle needed in situations such as when the master is sending set values to the slave.

The results of the tests with the real vehicles were very positive. Solving is very pleased with the new program structure and will now be able to do things with the vehicle software used in the air film movers that were not possible before this thesis. A few issues unrelated to this thesis were discovered during the tests with the real vehicles that may be addressed by Solving after this thesis. A few of these problems are mentioned as future work in the next chapter.

6. CONCLUSIONS AND FUTURE WORK

This document follows the same structure as the very common top-down design principle, meaning that the abstraction level is quite high in the beginning and very low at the end of the document. This document started with components and information that should in some extent be familiar to the majority of people and step by step lead the reader to a more detailed level of the topic in this thesis. As it was shown in this document, there are currently no other or at least no other public solutions for the problem this thesis is addressing.

It has been a long process getting from the first conceptual idea to a final solution tested with real equipment. The work of this thesis begun with a primitive idea gotten from Solving after which mathematical analysis done on pen and paper started immediately. One important thing noticed during this thesis that cannot be emphasized enough is how iterative a software project like this is. It has several times been necessary to jump from one phase backwards to an earlier phase to correct errors and regroup. When writing the SCL code of the FBs for instance, there were times when the construction of a certain vehicle type did not support the derived equations and it was then necessary to jump back to the research stage and do new mathematical analysis on pen and paper.

The outcome of this thesis are the FBs “ReferenceValueCalculation”, “DriveUnitSetValueCalculation” and the associated PLC data types created in Siemens STEP 7 (TIA Portal). The function of the FBs was verified both virtually with different simulations and physically by tests with real equipment. The FBs can be used to create program structures for vehicles with an unrestricted amount of wheels positioned freely in a Cartesian coordinate system. This document has explained the development process, the internal implementation and results of using the said FBs. The FBs, this document and documentation of the FBs have been handed over to Solving, so that they can start using the FBs in the software of the vehicles. Although the FBs were created for Siemens PLCs and Solving is currently in no need of using the program structure in any other type of logic controllers, let it be stated as a small side note that the FBs could most likely be converted to any other PLC brand as the Siemens FBD and SCL programming languages are closely related to the IEC 61131-3 standardized programming languages.

Testing of the new program structure was very successful, because of the possibility to use two air film movers in unison. By placing the air film movers at different orientations and distances to one another, several different wheel-layouts got tested. Although the principle should be the same for any other vehicle type, it is highly recommended that extra time is given for testing of the program structure when it is to be implemented in other vehicle types than the air film mover. Let it be mentioned that the tape guidance

navigation technique is being tested on the air film movers using the new program structure at the time when this chapter is being written. The results of those tests seem promising as well.

A few assumptions have been made in the beginning that may or may not restrict the use of the new program structure. The maximum range of a turning angle is for instance assumed to be [-180°, 180°]. There are however wheel types that allow the wheel to rotate freely around their axis. If Solving would need such wheels in the future, the SCL code of the FBs should need some modifications. Another assumption that was made is that the ICR should always be on the same side as the turning direction of the vehicle, meaning that if the vehicle is turning left, the ICR must also be on the left side of the vehicle. This restriction seems quite obvious, because the moving direction of the vehicle would change if the ICR would be on the opposite side compared to the turning direction, meaning that although the vehicle is told to move forward, it will move backwards and vice versa. As odd as it may seem there are however AGVs that actually need this quality. The program structure does support this quality, but it has been turned off because there were no vehicles to test the said quality with at the time.

Although the set value calculation of the set speed and turning angle of the wheel planes presented in this thesis worked perfectly, there were some problems with the control of the set values. The control is just to be clear a part of the vehicle software that has not been done in this thesis. Once the wheels found the correct set values, everything worked as intended, but the wheels were not synchronized during the transition from one set value to another. Because the goal of this thesis was to find a solution for the set value calculation only, the control part was not looked at as it would clearly have expanded the amount of work for this thesis. As long as the wheels are inside the same frame or under the same load, the control problem was negligible. A few suggestions about how the synchronization of the control values could be done have been given to Solving and engineers at Solving may or may not have to address this problem later on depending on what types of vehicles, what the requirements are and how the program structure will be used in the future. At this stage Solving does not however see any reason to develop the control of the set values.

The goals for this thesis were achieved and the results of the tests were in some extent even better than expected. The outcome of this thesis will enable Solving to do things that they were not able to do before this thesis. One of the first specific challenges Solving will overcome using the outcome of this thesis is production of the three wheeled air film movers.

This project has been very exciting to work with. Seeing end results in the physical form that originally started with just a conceptual idea is every engineers dream. Especially rewarding is that the results of this thesis can create new customer relationships, which means more work, more money and maybe even more jobs in the region near Solving.

REFERENCES

Audi. 2017. Audi Q7 all-wheel steering. [ONLINE] Available[27.09.2017]: https://www.audi-technology-portal.de/en/chassis/wheel-suspension-steering/audi-q7-all-wheel-steering-eng.

Bhishikar S., Vatsal G., Dalal N., Paarth M., Bhil S., Mehta A. (2014). Design and Simulation of 4 Wheel Steering System. International Journal of Engineering and Innovative Technology (IJEIT) Volume 3, Issue 12 pp. 351 – 367. Available[18.12.2016]: http://www.ijeit.com/Vol%203/Issue%2012/IJEIT1412201406_65.pdf

BMW. 2017. BMW Active steering. [ONLINE] Available[30.11.2016]: http://www.bmw.com/com/en/insights/technology/technology_guide/articles/mm_active_steering.html?content_type=index&source=/com/en/insights/technology/technology_guide/articles/integral_active_steering.html&article=mm_active_steering

Giancarlo G., Morello L., (2009). The Automotive Chassis, Volume 1: Components Design. Springer Netherlands. Steering System pp. 239-268. Available[30.11.2016]: http://link.springer.com/chapter/10.1007/978-1-4020-8676-2_4

Gitchens P. (1946). How Your Car Turns Corners. Popular Science. Bonnier Corporation. pp. 76-77. Available[27.09.2017]: https://books.google.fi/books?id=kiEDAAAAMBAJ&pg=PA76&redir_esc=y#v=onepage&q&f=false

Jazar R. (2008). Vehicle Dynamics: Theory and Application. Springer New York. Steering Dynamics pp. 379-454. Available[30.11.2016]: http://link.springer.com/chapter/10.1007/978-0-387-74244-1_7

Knowles D. (2011). Automotive Suspension & Steering Systems 5th edition Classroom manual Cengage learning. Available[30.11.2016]: http://engineeringstudymaterial.net/ebook/automotive-suspension-steering-classroom-manual/

Kong J., Pfeiffer M., Schildbach G., Borrelli F. (2015). Kinematic and Dynamic Vehicle Models for Autonomos Driving Control Design. 2015 IEEE Intelligent Vehicles Symposium (IV). pp. 1094-1099. Available[30.11.2016]: http://www.me.berkeley.edu/~frborrel/pdfpub/IV_KinematicMPC_jason.pdf

Lakkad S. (2004). Modeling and Simulation of Steering Systems for Autonomous Vehicles. Department of Mechanical Engineering, Florida State University. Available[30.11.2016]: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.360.9295&rep=rep1&type=pdf

Nissan. (2017). Nissan 4-wheel-active-steering. [ONLINE] Available[30.11.2016]: http://www.nissan-global.com/EN/TECHNOLOGY/OVERVIEW/4was.html

Novacek S. (2016). Wheelift Heavy Transporters Overcome Gravity on Earth to Help NASA Assembly Processes Reach for the Stars. Available[18.12.2016]: http://www.wheelift.com/resources-media/news/2016/03/01/wheelift-heavy-transporters-overcome-gravity-on-earth-to-help-nasa-assembly-processes-reach-for-the-stars

Rajamani R. (2006). Vehicle Dynamics and Control. Springer US. Electronic Stability Control. pp. 221-256. Available[30.11.2016]: http://link.springer.com/chapter/10.1007/0-387-28823-6_8

Siemens. (2014). Precise Synchronous Work for Huge Transportation Loads. [ONLINE] Available[29.09.2017]: https://www.siemens.com/press/en/presspicture/?press=/en/presspicture/innovationnews/2014/in20140402-01.htm&content[]=CT&content[]=I&content[]=DF&content[]=PD

Siemens. (2017). Programming Guideline and Programming Styleguide for S7-1200 and S7-1500. [ONLINE] Available[03.10.2017]: https://support.industry.siemens.com/cs/document/81318674/programming-guideline-and-programming-styleguide-for-s7-1200-and-s7-1500?dti=0&lc=en-WW

Singh A., Kumar A., Chaundhary R., Singh R. (2014). Study of 4 Wheel Steering Systems to Reduce Turning Radius and Increase Stability. Department of Mechanical Engineering, Delhi Technological University, Delhi, India. International Conference of Advance Research and Innovation pp. 96-102. Available[18.12.2016]: https://www.researchgate.net/publication/281450446_Study_of_4_Wheel_Steering_Systems_to_Reduce_Turning_Radius_and_Increase_Stability

Sironen H. (2015). Development of hydraulic valve systems for handling devices. Tampere University of Technology, Master of Science Thesis, 90 p. Available[31.07.2017]: https://dspace.cc.tut.fi/dpub/handle/123456789/22864

Snider J. M. (2009). Automatic Steering Methods for Autonomous Automobile Path Tracking. Robotics Institute, Carnegie Mellon University Pittsburgh Pennsylvania. Available[30.11.2016]: https://www.ri.cmu.edu/pub_files/2009/2/Automatic_Steering_Methods_for_Autonomous_Automobile_Path_Tracking.pdf

Solving. (2017). Information gathered from professionals working at Solving, partner pages to other companies accessed via Solving and Solving's homepage Available[18.12.2016]: https://www.solving.com/

Tadakuma K., Tadakuma R., Berengeres J. (2007). Development of holonomic omnidirectional Vehicle with "Omni-Ball": spherical wheels. 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 33-39. Available[19.12.2016]: http://ieeexplore.ieee.org/document/4399560/?arnumber=4399560

APPENDIX A: REFERENCE VALUE CALCULATION FB ERROR CODES

1101:

Explanation

The vehicle program is trying to activate the crabwise steering program, but it is disabled.

Troubleshooting

Enable the crabwise steering program in the “enabledPrograms” input in order to be able to activate it.

1102:

Explanation

The vehicle program is trying to activate the front wheel steering program, but it is disabled.

Troubleshooting

Enable the front wheel steering program in the “enabledPrograms” input in order to be able to activate it.

1103:

Explanation

The vehicle program is trying to activate the rear wheel steering program, but it is disabled.

Troubleshooting

Enable the rear wheel steering program in the “enabledPrograms” input in order to be able to activate it.

1104:

Explanation

The vehicle program is trying to activate the counter-phase steering program, but it is disabled.

Troubleshooting

Enable the counter-phase steering program in the “enabledPrograms” input in order to be able to activate it.

1105:

Explanation

The vehicle program is trying to activate the rotation mode steering program, but it is disabled.

Troubleshooting

Enable the rotation mode steering program in the “enabledPrograms” input in order to be able to activate it.

1106:

Explanation

The vehicle program is trying to activate crosswise mode, but it is disabled.

Troubleshooting

Enable crosswise mode in the “enabledPrograms” input in order to be able to activate it.

1107:

Explanation

The vehicle program is trying to activate the differential drive mode steering program, but it is disabled.

Troubleshooting

Enable the differential drive mode steering program in the “enabledPrograms” input in order to be able to activate it.

1120:

Explanation

Differential drive mode is not possible, changing to crabwise steering program.

Troubleshooting

If the vehicle width is zero, the vehicle cannot drive in the differential drive mode steering program. Change the vehicle width by changing the parameters “largestYCoordinate[mm]”, “smallestYCoordinate[mm]”, “largestXCoordinate[mm]” or “smallestXCoordinate[mm]”.

1200:

Explanation

Impossible to activate a steering program.

Troubleshooting

If this error code appears it has most likely got something to do with bad joystick inputs.

1300:

Explanation

Initialization process is running.

Troubleshooting

This error code will be displayed while the initialization process is running. As soon as the initialization process has finished the “ErrorInfo” parameter should go back to 0. If the “ErrorInfo” output stays at 1300 for a long time (minutes) there is an unexpected internal error in the FB.

3100:

Explanation

The selected steering configuration is undefined.

Troubleshooting

Change the value of the “steeringConfiguration” input to a valid value.

3200:

Explanation

The vehicle program has activated an undefined steering program.

Troubleshooting

Will not happen if the FB is used in the original intent. Can happen in situations where the SCL code of the FB has been modified.

4110:

Explanation

Left joystick input error (more than two digital signals active at once).

Troubleshooting

Please check the digital values of the left joystick. Are the inputs of the digital joystick values connected correctly to the FB? Is there a HW (hardware) related issue?

4120:

Explanation

Left joystick input error (UP and DOWN high at the same time).

Troubleshooting

Please check the digital values of the left joystick. Are the inputs of the digital joystick values connected correctly to the FB? Is there a HW related issue?

4130:

Explanation

Left joystick input error (RIGHT and LEFT high at the same time).

Troubleshooting

Please check the digital values of the left joystick. Are the inputs of the digital joystick values connected correctly to the FB? Is there a HW related issue?

4140:

Explanation

Left joystick input error (the horizontal analog value out of range).

Troubleshooting

Please check the analog values of the left joystick. Are the inputs of the analog joystick values connected correctly to the FB? Is there a HW related issue?

4150:

Explanation

Left joystick input error (the vertical analog value out of range).

Troubleshooting

Please check the analog values of the left joystick. Are the inputs of the analog joystick values connected correctly to the FB? Is there a HW related issue?

4160:

Explanation

Left joystick DI (Digital In) and AI (Analog In) mismatch (horizontal direction).

Troubleshooting

Please check the analog and digital values of the left joystick. Are the inputs of the joystick values connected correctly to the FB? Is there a HW related issue?

4170:

Explanation

Left joystick DI and AI mismatch (vertical direction).

Troubleshooting

Please check the analog and digital values of the left joystick. Are the inputs of the joystick values connected correctly to the FB? Is there a HW related issue?

4210:

Explanation

Right joystick input error (more than two digital signals active).

Troubleshooting

Please check the digital values of the right joystick. Are the inputs of the digital joystick values connected correctly to the FB? Is there a HW related issue?

4220:

Explanation

Right joystick input error (UP and DOWN high at the same time).

Troubleshooting

Please check the digital values of the right joystick. Are the inputs of the digital joystick values connected correctly to the FB? Is there a HW related issue?

4230:

Explanation

Right joystick input error (RIGHT and LEFT high at the same time).

Troubleshooting

Please check the digital values of the right joystick. Are the inputs of the digital joystick values connected correctly to the FB? Is there a HW related issue?

4240:

Explanation

Right joystick input error (the horizontal analog value out of range).

Troubleshooting

Please check the analog values of the right joystick. Are the inputs of the analog joystick values connected correctly to the FB? Is there a HW related issue?

4250:

Explanation

Right joystick input error (the vertical analog value out of range).

Troubleshooting

Please check the analog values of the right joystick. Are the inputs of the analog joystick values connected correctly to the FB? Is there a HW related issue?

4260:

Explanation

Right joystick DI and AI mismatch (horizontal direction).

Troubleshooting

Please check the analog and digital values of the right joystick. Are the inputs of the joystick values connected correctly to the FB? Is there a HW related issue?

4270:

Explanation

Right joystick DI and AI mismatch (vertical direction).

Troubleshooting

Please check the analog and digital values of the right joystick. Are the inputs of the joystick values connected correctly to the FB? Is there a HW related issue?

4310:

Explanation

Several steering programs activated at once.

Troubleshooting

Can happen for instance when the steering configuration 2 is used and more than one steering program is activated. Activate only one of the steering programs defined in the input “optionalSwitchforSteeringPrograms” at a time.

4320:

Explanation

Differential Drive mode and Rotation mode activated at the same time.

Troubleshooting

The differential drive mode and the rotation mode cannot be activated at the same time. Please check if there is an error in the inputs of the remote controller signals (“RCSignals”).

4410:

Explanation

Maximum desired heading out of range.

Troubleshooting

Check the turning angle limits of the set value calculation FBs and try to force a new run of the initialization process.

4500:

Explanation

Maximum vehicle speed out of range.

Troubleshooting

Please check that the “maxVehicleSpeed[mm/s]” is a positive floating-point number.

4600:

Explanation

Maximum analog joystick value out of range.

Troubleshooting

Please check that the maximum analog value of the “RCSignals” input is a positive floating-point number.

4710:

Explanation

Largest y-coordinate out of range.

Troubleshooting

Please check the size of the vehicle and redefine the parameter “largestYCoordinate[mm]”.

4720:

Explanation

Smallest y-coordinate out of range.

Troubleshooting

Please check the size of the vehicle and redefine the parameter “smallestYCoordinate[mm]”.

4730:

Explanation

Largest x-coordinate out of range.

Troubleshooting

Please check the size of the vehicle and redefine the parameter “largestXCoordinate[mm]”.

4740:

Explanation

Smallest x-coordinate out of range.

Troubleshooting

Please check the size of the vehicle and redefine the parameter “smallestXCoordinate[mm]”.

4800:

Explanation

Impossible to activate a steering program because all steering programs are disabled.

Troubleshooting

Please enable steering programs.

5100:

Explanation

Internal FB error.

Troubleshooting

This is an unexpected error that may occur if the FB is trying to do impossible things such as division by zero. To clear this error please contact support.

5210:

Explanation

ICR x-coordinate invalid value.

Troubleshooting

The value of the ICR x-coordinate is NaN (Not-a-Number) or there is an overflow or underflow. Check all inputs.

5220:

Explanation

ICR y-coordinate invalid value.

Troubleshooting

The value of the ICR y-coordinate is NaN (Not-a-Number) or there is an overflow or underflow. Check all inputs.

5230:

Explanation

Reference speed invalid value.

Troubleshooting

The value of the ICR x-coordinate is NaN (Not-a-Number) or there is an overflow or underflow. Check all inputs.

5300:

Explanation

Failed initialization process.

Troubleshooting

If the initialization process has failed the vehicle will not move. You have probably forgotten to enable steering programs or there is some other input error in either the “ReferenceValueCalculation” FB or the “DriveUnitSetValueCalculation” FB.

5310:

Explanation

Rotation mode has been disabled because it is not possible.

Troubleshooting

When the limiting turning angles that are inputs of the set value calculation FBs are too small, the rotation mode is not possible and will therefore be disabled automatically by the FB.

APPENDIX B: SET VALUE CALCULATION FB ERROR CODES

1100:

Explanation

Trying to turn, but the type is D.

Troubleshooting

A “D” type drive unit cannot turn. Are you sure you have the correct drive unit type?

1200:

Explanation

Trying to drive, but the type is S.

Troubleshooting

An “S” type drive unit cannot drive. Are you sure you have the correct drive unit type?

1300:

Explanation

Turning angle out of range, rotation mode not possible.

Troubleshooting

The angle flag will inform the “ReferenceValueCalculation” FB that the rotation mode is not possible and the “ReferenceValueCalculation” FB will disable rotation mode. This error code serves merely as an info-message.

4100:

Explanation

Input error, no active steering program.

Troubleshooting

There is no active steering program in the “references” input. Please check the “references” input.

4200:

Explanation

Input error, undefined drive unit type.

Troubleshooting

The value of the “type” input must be either 0, 1 or 2. Please check.

4300:

Explanation

X-coordinate out of range.

Troubleshooting

The size of the vehicle is unrealistically large. Please check the input “xCoordinate[mm]”.

4400:

Explanation

Y-coordinate out of range.

Troubleshooting

The size of the vehicle is unrealistically large. Please check input “yCoordinate[mm]”.

4510:

Explanation

Maximum speed out of range.

Troubleshooting

Please check the input “maxSpeed[mm/s]”. The input must be a positive floating-point number.

4610:

Explanation

Maximum positive turning angle out of range.

Troubleshooting

Please check the input “maxPositiveTurningAngle[deg]”. The value must be within the range [-180°, 180°].

4620:

Explanation

Maximum negative turning angle out of range.

Troubleshooting

Please check the input “maxNegativeTurningAngle[deg]”. The value must be within the range [-180°, 180°].

4800:

Explanation

Offset angle out of range.

Troubleshooting

Please check the input “offsetAngle[deg]”. The value must be within the range [-180°, 180°].

5100:

Explanation

Internal FB error.

Troubleshooting

This is an unexpected error that may occur if the FB is trying to do impossible things such as division by zero. To clear this error please contact support.

5210:

Explanation

Invalid turning angle.

Troubleshooting

The calculated “turningAngle[deg]” is invalid. Meaning that the value is NaN or there is an overflow or underflow. Please check all inputs.

5220:

Explanation

Invalid set speed.

Troubleshooting

The calculated “setSpeed[mm/s]” is invalid. Meaning that the value is NaN or there is an overflow or underflow. Please check all inputs.